Міністерство освіти і науки України
Тернопільський національний технічний університет імені
Івана Пулюя

Кафедра комп'ютерних систем та мереж

Методичні вказівки

до виконання лабораторних робіт

з дисципліни:

«Моделювання систем»

для студентів денної форми навчання за напрямом 6.050102
"Комп'ютерна інженерія"

Тернопіль
2015

Методичні вказівки розроблені у відповідності з навчальним планом напрямку 6.050102 „Комп'ютерна інженерія ”

Укладачі: асисистент кафедри комп'ютерних систем та мереж Луцик Н.С.

Рецензент: доцент кафедри комп'ютерних наук Литвиненко Я.В.

Затверджено на засіданні кафедри комп'ютерних систем та мереж, протокол №__ від «__» вересня 2015 р.

Схвалено та рекомендовано до друку методичною комісією факультету комп'ютерно-інформаційних систем і програмної інженерії Тернопільського національного технічного університету імені Івана Пулюя, протокол № _____ від «____» вересня 2015 р.

Посібник складений з врахуванням методичних розробок інших вищих закладів освіти, а також матеріалів літературних джерел, перелічених в списку.

ЗМІСТ

ВСТУП

Ціллю даного лабораторного практикуму є практичне освоєння методів математичного моделювання в різних програмних середовищах. В інженерних дисциплінах математичний опис об'єктів моделювання грає величезну роль. Тому важливо не тільки теоретично вивчити методи моделювання та умови їх застосування, а й апробувати їх на реальних даних з типових об'єктів моделювання.

В даному лабораторному практикумі розглядається моделювання випадкових сигналів, псевдовипадкових величин, моделювання систем масового обслуговування із одним пристроєм обслуговування, моделювання поведінки обчислювальних структур з використанням апарату мереж Петрі та вивчення поведінки дискретних динамічних систем на прикладні клітинних автоматів.

Виконання завдань передбачає, що студенти освоїли теоретичний матеріал, володіють функціями та операторами MathCad та MATLAB, готові до їх реалізації на комп'ютері.

Методичні вказівки призначені для студентів денної форми за напрямком підготовки 6.050102 «Комп'ютерна інженерія».

Лабораторна робота №1

Тема. Моделювання випадкових чисел

Мета роботи. Вивчити методи та алгоритми моделювання випадкових величин.

Теоретичні відомості

1. Основні імовірнісні поняття

Випадковим досвідом або ***експериментом*** називається процес, при якому можливі різні результати, так що не можна заздалегідь передбачити, який буде результат. Величина $X=\{x_i\}=x1, x2, ..., x_n$, що є результатом випадкового досвіду, називається ***випадковою величиною***. Мінливість результату такого досвіду може бути пов'язане з наявністю випадкових помилок вимірювань або зі статистичною природою самої вимірюваної величини (наприклад, процес розпаду радіоактивної речовини). Будемо позначати окремі значення, які приймає випадкова величина (не обов'язково чисельні), як x_i, де $i = 1, 2, ...,n$. Будь-яка функція від x_i буде також випадковою величиною.

Під ***моделюванням*** випадкової величини X прийнято розуміти процес отримання на ЕОМ її вибіркових значень $x_1, ..., x_n$. На практиці використовуються три основних способи генерації випадкових чисел:

-*Табличний (файловий)* - введення таблиць рівномірно розподілених випадкових чисел у зовнішню або оперативну пам'ять ЕОМ;

- *Апаратний (фізичний)* - використання спеціального пристосування до ЕОМ - "датчика" випадкових чисел, який формує випадкові величини шляхом фізичного моделювання деяких випадкових процесів (випромінювання радіоактивних джерел, шумів електронних ламп та ін);

-*Алгоритмічний (програмний)* - використання псевдовипадкових послідовностей, що реалізуються програмним генератором випадкових чисел. ***Псевдовипадковими*** числами називаються числа, що виробляються ЕОМ рекурентним способом за спеціальними алгоритмами, коли кожне наступне число x_i виходить з попередніх в результаті застосування деяких арифметичних і логічних операцій. Така послідовність чисел задовольняє відомими критеріями випадковості, хоча вхідні в цю послідовність числа залежні між собою. Одним з недоліків цього методу є періодичність утворених програмним способом псевдовипадкових чисел, але для ряду завдань, які потребують великої кількості випадкових чисел, довжина періоду є достатньою.

Перший спосіб. При вирішенні задачі без застосування ЕОМ найчастіше використовують таблиці випадкових чисел. Таблиці отримують за допомогою спеціальних приладів (типу рулетки) і заносять в пам'ять ЕОМ, використовуються по мірі необхідності.

У таблицях випадкових чисел випадкові цифри імітують значення дискретної випадкової величини з рівномірним розподілом:

x_i	0	1	2	3	…	9
p_i	0,1	0,1	0,1	0,1	…	0,1

При складанні таких таблиць виконується вимога, щоб кожна з цих цифр від 0; 1; ...;9 зустрічалася приблизно однаково часто і незалежно від інших з ймовірністю $p_i = 0,1$.

Найбільша з опублікованих таблиць випадкових чисел містить 1000000 цифр. Таблиці випадкових чисел вимагають ретельної перевірки за допомогою спеціальних статистичних тестів.

Основний недолік - необхідність у пам'яті досить великої ємності, що утруднює рішення "великих" завдань, тим більше що перевага "випадкових" таблиць перед "псевдовипадковими" числами, одержуваними алгоритмічно, ніким не було доведено.

У другому способі використовуються апаратні датчики, засновані на деяких фізичних процесах, випадкових за своєю природою (шуми в електронних і напівпровідникових приладах, процеси при радіоактивному розпаді і т.п.).

Або ж при вирішенні завдань на ЕОМ для вироблення випадкових чисел, рівномірно розподілених в інтервалі (0; 1), можуть застосовуватися генератори випадкових чисел. Дані генератори перетворюють результати випадкового фізичного процесу в двійкові числа. В якості випадкового фізичного процесу зазвичай використовують власні шуми (випадковим чином змінюється напруга).

Основні недоліки - неможливість повторного отримання однієї і тієї ж послідовності випадкових величин для перевірочних розрахунків і неможливість гарантувати постійну надійну роботу датчика.

Третій спосіб. Отримання псевдовипадкових чисел з рівномірним законом розподілу полягає у виробленні псевдовипадкових чисел.

Псевдовипадкові числа - це числа, отримані з будь-якої формулою і імітують значення випадкової величини. Під словом "імітують" мається на увазі, що ці числа задовільняють ряд тестів так, як якщо б вони були значеннями цієї випадкової величини.

Перший алгоритм для отримання псевдовипадкових чисел запропонував Дж. Нейман. Це так званий метод середини квадратів, який полягає в наступному:

$x_0 = 0,9876, x_0^2 = 0,97\underline{5353}76,$

$x_1 = 0,5353, x_1^2 = 0,28\underline{6546}09,$

$x_2 = 0,6546$

Метод середин квадратів фон Неймана є порівняно бідним джерелом випадкових чисел, тому що послідовність прагне увійти у звичну колію, тобто короткий цикл повторюваних елементів.

Переваги методу псевдовипадкових чисел:

1.На отримання кожного випадкового числа витрачається кілька простих операцій, так що швидкість генерування випадкових чисел має той самий порядок, що і швидкість роботи ЕОМ.

2. Малий обсяг пам'яті ЕОМ для програмування.

3. Будь-яке з чисел легко відтворити.

4.Якість генеруються випадкових чисел досить перевірити один раз.

Випадкові величини бувають дискретні і безперервні, одномірні (залежні від однієї змінної) або багатовимірні (залежні від двох і більше змінних).

Дискретною випадковою величиною називається така величина, число можливих значень якої або кінцеве, або нескінченне рахункове.

Неперервною випадковою величиною називається така величина, можливі значення якої безперервно заповнюють деякий інтервал (кінцевий чи нескінченний) числової осі.

Повною характеристикою випадкової величини X з імовірнісної точки зору є її ***закон розподілу***, тобто заданий в тій чи іншій формі зв'язок між можливими значеннями випадкової величини і ймовірностями їх появи.

Універсальною формою закону розподілу (безперервних і дискретних величин) є ***функція розподілу ймовірностей*** - це така функція F (x), значення якої в точці x рівне ймовірності (P) того, що при проведенні досвіду значення випадкової величини X виявиться менше, ніж x:

$F(x) = P(X < x)$.

Основні властивості функції розподілу ймовірностей наступні:

1) числові значення укладені в межах $0 \le F(x) \le 1$;
2) якщо $x1 \le x2$, то $F(x1) \le F(x2)$, тобто $F(x)$ - неспадаюча функція;
3) $F(x) \to 0$ при $x \to -\infty$, $F(x) \to 1$ при $x \to \infty$.

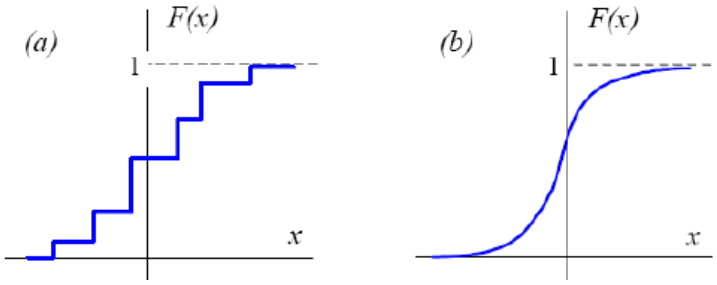


Рисунок 1. Графічне зображення функції розподілу ймовірностей

Якщо випадкова величина дискретна, то її функція розподілу являє собою східчасту функцію (рисунок 1a), а у безперервних випадкових величин функція розподілу також неперервна (малюнок 1b).

Функцію розподілу ймовірностей F (x) неперервної випадкової величини можна представити у вигляді інтеграла від деякої невід'ємної функції f (x):

$$F(x) = \int_{-\infty}^{x} f(u)du\,.$$

Функція f (x) називається ***щільністю розподілу ймовірності***. Основні властивості щільності ймовірності такі:

1) $f(x) = F'(x) = \dfrac{dF(x)}{dx}$; $F(x) = \int_{-\infty}^{x} f(u)du$;

2) $\int_{-\infty}^{\infty} f(u)du = 1$;

3) щільність ймовірності пропорційна ймовірності події $(x \le X \le x + dx)$.

Крім закону розподілу, випадкову величину характеризують значеннями деяких параметрів, що визначають найбільш суттєві особливості її розподілу. Найбільш часто використовуваними параметрами розподілу є математичне

очікування чи середнє значення випадкової величини, а також дисперсія випадкової величини.

1.1 Параметри випадкової величини

Основне призначення числових характеристик випадкової величини полягає в тому, щоб в стислій формі висловити найбільш суттєві особливості того чи іншого розподілу.

Математичним сподіванням чи ***середнім значенням*** дискретної випадкової величини називається сума всіх можливих значень x_i випадкової величини X, помножених на відповідні ймовірності:

$$M\{X\} \equiv \bar{x} = \sum_{i=1}^{n} x_i P(X = x_i) = \sum_{i=1}^{n} x_i P_i ,$$

$$M\{X^2\} = \sum_{i=1}^{n} (x_i)^2 P_i ,$$

$$(M\{X\})^2 = \left(\sum_{i=1}^{n} x_i P_i \right)^2 .$$

Зауважимо, що *x* є не випадковою, а певною, детермінованою величиною.

Так як функція від випадкової величини є також випадковою величиною, то математичне сподівання функції $Y = H(X)$ визначається наступним чином:

$$M\{H(X)\} = \sum_{i=1}^{n} H(x_i) P(X = x_i) = \sum_{i=1}^{n} H(x_i) P_i .$$

Для неперервних випадкових величин будемо мати:

$$M\{X\} \equiv \bar{x} = \int_{-\infty}^{\infty} x f(x) dx \text{ і } M\{H(X)\} \equiv \bar{x} = \int_{-\infty}^{\infty} H(x) f(x) dx .$$

Важливою характеристикою відхилення або розподілу випадкової величини від її середнього значення є ***дисперсія*** випадкової величини, що визначається як математичне сподівання квадрата відхилення випадкової величини X від свого середнього значення:

$$D\{X\} \equiv \sigma^2(X) = M\left\{ (X - M\{X\})^2 \right\} = M\left\{ (X - \bar{x})^2 \right\} \equiv M\{X^2\} - (M\{X\})^2 .$$

Додатній квадратний корінь з дисперсії $\sigma = +\sqrt{\sigma^2(X)}$ називається ***стандартним*** або ***середньоквадратичним відхиленням***. Середньоквадратичне відхилення кількісно показує, наскільки сильно значення випадкової величини X розподілені навколо середнього значення x.

2.3 Моделювання дискретних випадкових величин

Розглянемо дискретну випадкову величину X, приймаючу n значень x_1, x_2, ..., x_n з ймовірностями P_1, P_2 ,..., P_n. Ця величина задається таблицею розподілу

$$X = \begin{pmatrix} x_1 & x_2 & \dots & x_n \\ P_1 & P_2 & \dots & P_n \end{pmatrix} \qquad \sum_{i=1}^{n} P_i = 1$$

або

X	x_1	x_2	x_3	…	x_n
P	P_1	P_2	P_3	…	P_n

Для моделювання такої дискретної випадкової величини розбивають відрізок [0; 1] на n послідовних відрізків Δ1, Δ2 ,..., Δn, довжини яких

дорівнюють відповідним ймовірностями P_1, P_2, ..., P_n. Тоді довжини відрізків будуть рівні:

Довжина $\Delta_1 = P_1 - 0 = P_1$

Довжина $\Delta_2 = (P_1 + P_2) - P_1 = P_2$

.....

Довжина $\Delta_n = 1 - (P_1 + P_2 + ... + P_{n-1}) = P_n$

Видно, що довжина часткового інтервалу з індексом і дорівнює ймовірності P з тим же індексом. Довжина $\Delta_i = P_i$. Отримують випадкову величину R, рівномірно розподілену в інтервалі (0; 1). Таким чином, при попаданні випадкового числа r_i в інтервал Δ_i випадкова величина X приймає значення x_i з імовірністю P_i.

Теорема: *якщо кожному випадковому числу ri(0≤ ri <1), яке потрапило в інтервал, поставити у відповідність можливе значення x_i, то розігрувана величина буде мати заданий закон розподілу.*

Розглянемо алгоритм моделювання дискретних випадкових величин:

1. Потрібно розбити інтервал [0;1) на n часткових інтервалів:

$\Delta_1 – (0; P_1)$, $\Delta_2 – (P_2; P_1 + P_2)$, …, $\Delta_n – (P_1 + P_2 + … + P_{n-1}; 1)$.

2. Вибрати (наприклад, з таблиці випадкових чисел, або в комп'ютері) випадкове число r_i з інтервалу [0; 1). Якщо r_i потрапило в інтервал, то модельована дискретна випадкова величина прийняла можливе значення x_i.

2.4 Моделювання неперервних випадкових величин

Може виявитися так, що алгоритм чисельного вирішення зазначеного рівняння буде досить складним або вимагати помітних витрат часу на обчислення. Тоді можуть бути використані інші методи генерування випадкових величин. Серед цих методів відзначимо ***метод виключення***.

Суть методу виключення (або методу Неймана) полягає в наступному: Нехай випадкова величина X визначена на кінцевому інтервалі (a; b) і щільність її розподілу обмежена, так що $f(x) \le M$. Тоді, використовуючи пару рівномірно розподілених на інтервалі (0;1) випадкових чисел R, здійснюємо наступні дії для розіграшу (моделювання) значення X:

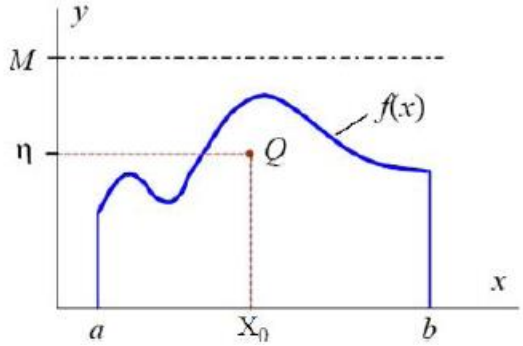


Рисунок 4. Графічне зображення методу Неймана

1. Розігруємо два значення r1 і r2 випадкової величини R і будуємо випадкову точку *Q* з координатами (див. рис. 4):

$X_0 = a + r_1 \cdot (b - a)$, $\eta = r_2 \cdot M$.

2.Якщо $\eta > f(X_0)$, то пару значень (r1, r2) відкидаємо і переходимо до пункту 1; інакше приймаємо $X = X_0$.

Таким чином, визначаються координати випадкової точки $Q(X_0,\eta)$ і, якщо точка буде під кривою *f(x)*, то абсциса цієї точки приймається як значення випадкової величини $X=X_0=a+r_1\cdot(b-a)$ з щільністю розподілу *f(x)*. В іншому випадку точка відкидається, визначаються координати наступної точки, і все повторюється.

Існують і інші численні способи формування випадкових величин з різними певними законами розподілу.

3. Перевірка гіпотези про закон розподілу методом гістограм

Нехай у результаті експерименту отримано n значень $x_1, x_2, ..., x_n$ випадкової величини X і всі вони укладені в межах $a < x_i < b$.

Суть перевірки по гістограмі зводиться до наступного:

а) Висувається гіпотеза про рівномірність розподілу чисел в інтервалі (0;1).

б) Потім інтервал (a;b) розбивається на L (будь-яке число, не дуже велике і не занадто мале) рівних підінтервалів довжиною Δj, тоді при генерації послідовності $\{x_i\}$ кожне з чисел x з імовірністю $p_j = 1/L$, $j=\overline{1,L}$, потрапляє в один з підінтервалів. Всього в кожен j-й підінтервал потрапляє V_j чисел послідовності $\{x_i\}$, $i=\overline{1,n}$, причому $\nu=\sum_{j=1}^{L}\nu_j$. Відносна частота потрапляння випадкових чисел послідовності $\{x_i\}$ в кожен з підінтервалів буде дорівнює ν_j/n.

в) Над кожним з підінтервалів розбиття будується прямокутник, площа якого дорівнює частоті потрапляння $\{x_i\}$ в цей підінтервал. Висота кожного прямокутника дорівнює частоті, поділеній на Δj. Отриману ступінчасту лінію називають ***гістограмою***.

г) Вид відповідної гістограми представлений на рисунку 5, де пунктирна лінія відповідає теоретичному значенню p_j, а суцільна - експериментальному ν_j/n. Очевидно, що якщо числа $\{x_i\}$ належать псевдовипадковій рівномірно розподіленій послідовності, то при досить великих n експериментальна гістограма (ламана лінія на рис. 5) наблизиться до теоретичної прямою $p_j=1/L$.

Гістограма служить наближенням до невідомої щільності випадкової величини X. Площа гістограми, укладена між x_i і x_i+1, дає наближене значення ймовірності $P\{x_i<X<x_i+1\}$.

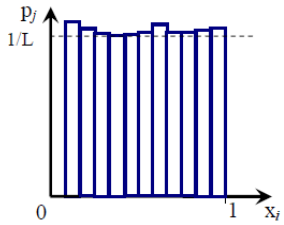


Рисунок 5. Перевірка рівномірності послідовності

Порядок виконання роботи

1. Згенерувати послідовність з n00 псевдовипадкових чисел, результат вивести на екран. (n – порядковий номер студента в журналі)

1.1. Оцінити математичне сподівання отриманої послідовності.

1.2.Оцінити дисперсію отриманої послідовності

1.3. Побудувати таблицю 1 (кількість L підінтервалів не менше 10), частотну таблицю вивести на екран.

Таблиця 1 - Частотна таблиця

Інтервал	Кількість СВ (частота попадань), які випали в даний інтервал	Відносна частота потрапляння
Δ_1	ν_1	ν_1/n
Δ_2	ν_2	ν_2/n
…	…	…
Δ_L	ν_L	ν_L/n
	$\sum$ кол-во СВ	

1.4. Перевірити гіпотезу про закон розподілу методом гістограм, побудувати гістограму, вивести її на екран.

2. Змоделювати дискретну випадкову величину, задану таблицею 2, результат вивести на екран.

2.1.Оцінити математичне сподівання отриманої дискретної випадкової величини, результат вивести на екран.

2.2. Оцінити дисперсію отриманої дискретної випадкової величини, результат вивести на екран.

2.3. Побудувати частотну таблицю, вивести її на екран.

2.4 Перевірити гіпотезу про закон розподілу методом гістограм, побудувати гістограму, вивести її на екран.

Таблиця 2 - Таблиця розподілів

Варіант	Таблиця розподілу							
1	x_i	5	7	17	19	21	25	55
	p_i	0.01	0.05	0.3	0.3	0.3	0.02	0.02
2	x_i	1	3	7	10	15	18	23
	p_i	0.1	0.05	0.02	0.05	0.25	0.33	0.2
3	x_i	2	3	5	12	21	33	44
	p_i	0.1	0.15	0.2	0.05	0.02	0.33	0.15
4	x_i	5	8	13	16	21	24	29
	p_i	0.1	0.02	0.25	0.15	0.35	0.03	0.1
5	x_i	2	3	5	8	11	15	20
	p_i	0.1	0.15	0.25	0.05	0.05	0.3	0.1
6	x_i	1	8	17	23	37	42	50
	p_i	0.01	0.15	0.05	0.25	0.5	0.02	0.02
7	x_i	1	4	12	16	25	33	37
	p_i	0.05	0.25	0.25	0.15	0.13	0.1	0.07
8	x_i	1	10	15	23	29	38	42
	p_i	0.02	0.05	0.1	0.28	0.23	0.22	0.1
9	x_i	2	3	7	12	19	23	30
	p_i	0.04	0.15	0.2	0.25	0.2	0.15	0.01
10	x_i	1	5	7	14	21	26	31
	p_i	0.34	0.28	0.16	0.15	0.05	0.01	0.01

11	x_i	3	5	8	14	27	29	35
	p_i	0.02	0.07	0.1	0.19	0.19	0.2	0.23
12	x_i	7	16	28	33	39	46	56
	p_i	0.01	0.05	0.07	0.1	0.17	0.25	0.35
13	x_i	5	6	8	13	19	26	36
	p_i	0.05	0.07	0.2	0.23	0.17	0.23	0.05
14	x_i	3	9	18	23	29	27	45
	p_i	0.05	0.14	0.2	0.22	0.17	0.14	0.08
15	x_i	13	16	28	33	39	47	52
	p_i	0.08	0.14	0.25	0.16	0.25	0.09	0.03
16	x_i	1	6	8	13	19	24	27
	p_i	0.09	0.1	0.21	0.17	0.23	0.15	0.05
17	x_i	4	6	10	14	16	20	24
	p_i	0.04	0.1	0.1	0.27	0.33	0.13	0.03
18	x_i	2	6	12	16	22	26	32
	p_i	0.02	0.14	0.24	0.27	0.2	0.1	0.03
19	x_i	3	6	9	13	19	27	31
	p_i	0.04	0.12	0.22	0.28	0.2	0.1	0.04
20	x_i	1	3	8	11	19	29	33
	p_i	0.02	0.26	0.18	0.32	0.16	0.02	0.04

Контрольні запитання

1. Що називають випадковим досвідом?
2. Що розуміють під моделюванням випадкової вечини?
3. Дайте визначення псевдовипадкових чисел.
4. Які переваги використання методу псевдовипадкових чисел?
5. Дайте означення функції та щільності розподілу?
6. Як відбувається процес моделювання непевних випадкових величин?
7. Проаналізуйте процес отримання рівномірно роз приділених випадкових чисел.

Лабораторна робота № 2

Тема: Моделювання системи масового обслуговування з одним пристроєм обслуговування

Мета роботи: Детально розглянути моделювання системи масового обслуговування (СМО) з одним пристроєм обслуговування. Це банк з одним працівником - касиром. Реалізувати програмно, відповідно до свого варіанту, СМО, визначити завдання, показати схеми модельованої системи після виникнення кожної події, описати результати моделювання, умови зупинки як спосіб закінчення моделювання.

Теоретичні відомості

Системи масового обслуговування та їхні характеристики

Із системами масового обслуговування (СМО) ми зустрічаємось повсякчас. Кожному з нас доводилось чекати обслуговування в черзі (у магазині, на автозаправці, в бібліотеці, кав'ярні тощо). Аналогічні ситуації виникають, коли треба скористатися телефонним зв'язком або виконати свою програму на комп'ютері. Будь-яке виробництво теж можна уявити як послідовність систем обслуговування. До типових систем обслуговування належать також ремонтні і медичні служби, транспортні системи, аеропорти, вокзали тощо.

Особливого значення набули такі системи у процесах інформатики. Це передусім комп'ютерні системи, мережі передавання інформації, операційні системи, бази і банки даних. Системи обслуговування відіграють значну роль у повсякденному житті. Досвід моделювання різних типів дискретних систем свідчить про те, що приблизно 80% цих моделей ґрунтуються на СМО.

Систему масового обслуговування загалом можна уявити як сукупність послідовно пов'язаних між собою вхідних потоків вимог на обслуговування (потоків замовлень), черг, каналів обслуговування і потоків обслужених замовлень. Будь-який пристрій, який безпосередньо обслуговує замовлення, називають каналом обслуговування. Системи масового обслуговування за наявності тої чи іншої ознаки можна класифікувати так:

1. За характером надходження замовлень у систему: на системи з регулярним і випадковим потоками замовлень. Якщо кількість замовлень, які надходять у ситему за одиницю часу (інтенсивність потоку), стала або є заданою функцією часу, то маємо систему з регулярним потоком замовлень, в іншому разі – з випадковим. Випадковий потік замовлень може бути стаціонарним або нестаціонарним. Якщо параметри потоку замовлень не залежать від розташування інтервалу часу, який розглядають, на осі часу, то маємо стаціонарний потік замовлень, в протилежному випадку – нестаціонарний. Наприклад, якщо кількість покупців, які приходять до магазину, не залежить від часу доби, то потік замовлень (покупців) – стаціонарний.

2. *За кількістю замовлень, які надходять за одиницю часу:* на системи з ординарним і неординарним потоками замовлень. Якщо ймовірність надходження двох або більше замовлень в один момент часу дорівнює нулеві або настільки мала, що нею можна знехтувати, то маємо систему з ординарним

потоком замовлень. Наприклад, потік літаків, які прибувають на злітну смугу аеродрому (ЗС), можна вважати ординарним, оскільки ймовірність надходження двох і більше літаків до каналу обслуговування (ЗС) в один і той самий момент часу дуже мала.

3. *За зв'язком між замовленнями*: на системи без післядії від замовлень, які надійшли, і з післядією. Якщо ймовірність надходження замовлень у систему в деякий момент часу не залежить від того, скільки вимог уже надійшло до системи, тобто не залежить від передісторії процесу, який вивчають, то ми маємо задачу без післядії, у протилежному випадку – з післядією. Прикладом задачі з післядією може слугувати потік студентів на складання заліку викладачеві.

4. *За характером поведінки замовлень у системі*: з відмовами, з обмеженим очікуванням і з очікуванням без обмеження: – якщо нове замовлення, яке прибуло на обслуговування, застає усі канали обслуговування уже зайнятими і покидає систему, то маємо систему з відмовами. Замовлення може покинути систему і тоді, коли черга досягла певних розмірів. Якщо ракета супротивника з'являється в час, коли всі протиракетні пристрої обслуговують інші ракети, то вона без проблем залишає зону обслуговування; – якщо нове замовлення, яке прибуло на обслуговування, застає усі канали обслуговування зайнятими і стає у чергу, але перебуває у ній обмежений час і, не дочекавшись обслуговування, покидає систему, то маємо систему з обмеженим очікуванням. Прикладом такого „нетерплячого" замовлення може бути самоскид із цементним розчином. Якщо час очікування великий, то щоб запобігти затвердненню розчину, він може бути розвантажений в іншому місці; – якщо нове замовлення, яке прибуло на обслуговування, заставши усі канали обслуговування зайнятими, змушене очікувати своєї черги до того часу, поки не буде обслужене, то маємо систему з очікуванням без обмеження. Приклад: літак, який перебуває на аеродромі до того часу, поки не звільниться злітна смуга.

5. *За способом вибору замовлень на обслуговування:* з пріоритетом, за часом надходження, випадково, останнього обслуговують першим. Іноді в такому випадку кажуть про дисципліну обслуговування:

– якщо система масового обслуговування охоплює кілька категорій замовлень і з певних міркувань необхідно дотримуватись різного підходу до їхнього відбору, то маємо систему з пріоритетом. Зокрема, під час надходження виробів на будмайданчик, перш за все монтують ті, які необхідні у цей момент;

– якщо канал, який звільнився, обслуговує замовлення, яке раніше за інших надійшло до системи, то маємо систему з обслуговуванням замовлень за часом надходження. Це найпоширеніший клас систем. Наприклад, покупця, який підійшов до продавця першим, обслуговують раніше за інших. Цей спосіб вибору замовлень на обслуговування застосовують там, де внаслідок технічних, технологічних або організаційних умов замовлення не можуть випереджати одне одного;

– якщо замовлення з черги надходять до каналу обслуговування у випадковому порядку, то маємо систему з випадковим вибором замовлень на обслуговування. Приклад: вибір слюсарем-сантехніком одного з декількох

замовлень на усунення несправностей, які надійшли від мешканців. Вибір тут, зазвичай, визначають місцезнаходженням самого слюсаря: він надасть перевагу замовленню мешканця, який перебуває від нього найближче, якщо інші чинники не визначають вибору;

– останнього обслуговують першим. Цей спосіб вибору вимог на обслуговування використовують у тих випадках, коли зручніше й економніше брати на обслуговування замовлення, яке найпізніше надійшло до системи. Зокрема, якщо будівельні вироби складені один на одному, то зручніше спочатку брати виріб, який надійшов останнім.

6. *За характером обслуговування замовлень:* на системи з детермінованим і випадковим часом обслуговування. Якщо інтервал часу між моментами надходження замовлення до каналу обслуговування і моментом виходу замовлення з цього каналу є сталим, то йдеться про систему з детермінованим часом обслуговування, в іншому разі – з випадковим.

7. *За кількістю каналів обслуговування:* на одноканальні і багатоканальні системи. Наприклад, для зведення будинку можна використати один будівельний кран (один канал обслуговування) або декілька (багато каналів) для обслуговування виробів, які прибувають на будову.

8. *За кількістю етапів обслуговування:* на однофазні і багатофазні системи. Якщо канали обслуговування розташовані послідовно, і вони неоднорідні, оскільки виконують різні операції обслуговування, то йдеться про багатофазну систему масового обслуговування. Прикладом такої системи може бути обслуговування автомобілів на станції технічного обслуговування (миття, діагностування тощо).

9. *За однорідністю замовлень, які надходять на обслуговування:* на системи з однорідними і неоднорідними потоками замовлень. Наприклад, якщо для розвантаження прибувають фургони однакової вантажомісткості, то такі замовлення називають однорідними, якщо різної – то неоднорідними.

10. *За обмеженістю потоку замовлень:* на замкнені і розімкнені системи. Якщо потік замовлень обмежений і замовлення, які покинули систему, через деякий час до неї повертаються, то маємо замкнену систему, в протилежному випадку – розімкнену. Прикладом замкненої ситеми може слугувати бригада робітників, які налагоджують станки в ткацькому цеху. З метою скорочення запису для позначення будь-якої однофазної СМО використовують систему кодування A/B/C/D/E, де на місці латинської літери ставлять відповідні характеристики системи:

A – закон розподілу інтервалів між надходженнями замовлень. Найчастіше використовують такі закони розподілу: показниковий (M), ерлангівський (E), гіперекспоненціальний (H), ґамма-розподіл (Г), детермінований (D). Для позначення довільного характеру розподілу використовують символ G;

B – закон розподілу часу обслуговування в каналах СМО. Тут використовують такі самі позначення, як і для розподілу інтервалів між надходженнями замовлень;

C – кількість каналів обслуговування. Тут використовують такі позначення: для одноканальних систем записують 1, для багатоканальних l (кількість каналів);

D – кількість місць у черзі. Якщо кількість місць у черзі необмежена, то це позначення можна не використовувати. Для скінченної кількості місць у черзі в загальному випадку приймають позначення r або n (кількість місць);

E – дисципдіна обслуговування. Найчастіше використовують такі варіанти системи обслуговування: FIFO (першим прийшов – першим вийшов), LIFO (останнім прийшов – першим вийшов), RANDOM (випадковий порядок обслуговування). За дисципліни FIFO це позначення можна не використовувати.

Приклади позначень:

M/M/1 – СМО з одним каналом обслуговування, нескінченною чергою, показниковими законами розподілу інтервалів часу між надходженнями замовлень і часу обслуговування та дисципліною обслуговування FIFO;

E/H/l/r/LIFO – СМО з кількома каналами обслуговування, скінченною чергою, ерлангівським законом розподілу інтервалів часу між надходженнями замовлень, гіперекспоненціальним розподілом часу обслуговування та дисципліною обслуговування LIFO;

G/G/l – СМО з кількома каналами обслуговування, нескінченною чергою, довільними законами розподілу інтервалів часу між надходженнями замовлень і часу обслуговування та дисципліною обслуговування FIFO.

Вивчення або задання потоку замовлень, механізму (кількості каналів, тривалості обслуговування тощо) та дисципліни обслуговування дає підстави для побудови моделі системи.

Завдання

Програмно реалізувати роботу касового залу банку і показати графічно стан черги клієнтів і зайнятість касира.

Умови постановки задачі:

1. Касир - 1 чол.
2. Час приходу клієна моделюється за експоненціальним розподілом з середнім часом **tсер.прих.** для інтервалів між надходженнями.
3. Час обслуговування клієнта моделюється за експоненціальним розподілом з середнім часом **tсер.обслуг.**
4. Обмеження на роботу касового залу: **tроб.залу.**

Необхідно розрахувати:

1. Середнє число заявок, що обслуговуються за одиницю часу
2. Інтенсивність потоку заявок
3. Середню частку заявок, що обслуговуються системою
4. Інтенсивність потоку обслуговувань

Варіанти завдань

№ варіанту	tроб.залу. з_год. до_год.	tсер.прих., хв	tсер.обслуг., хв
1	1-14	1	0.1
2	2-15	2	0.2
3	3-16	3	0.3
4	4-17	4	0.4
5	5-18	5	0.5
6	6-19	6	0.6
7	7-20	7	0.7
8	8-21	8	0.8
9	9-22	9	0.9
10	10-23	1	0.1
11	11-24	2	0.2
12	12-1	3	0.3
13	13-2	4	0.4
14	14-3	5	0.5
15	15-4	6	0.6
16	16-5	7	0.7
17	17-6	8	0.8
18	18-7	9	0.9
19	19-8	1	0.1
20	20-9	2	0.2
21	21-10	3	0.3
22	22-11	4	0.4
23	23-12	5	0.5
24	24-13	6	0.6
25	6-11	7	0.7

Контрольні питання

1. Що є вхідними даними системи масового обслуговування?
2. Які є критерії оцінки роботи системи?
3. Що є проміжними даними системи?
4. Охарактеризуйте що є елементами та зв’язками системи?
5. До якого класу систем відноситься система масового обслуговування?
6. Який принцип обробки вимог у системі?

Лабораторна робота №3

Тема. Використання апарату мереж Петрі для моделювання поведінки обчислювальних структур

Мета. Вивчення матричних способів представлення мереж Петрі (МП) і методів дослідження МП-моделей на основі матричних рівнянь.

Короткі теоретичні відомості

Мережа Петрі складається з 4 компонентів, які й визначають її структуру:
- безліч позицій P,
- безліч переходів T,
- вхідна функція I,
- вихідна функція O.

Для ілюстрації понять теорії мереж Петрі набагато більш зручно графічне представлення мережі Петрі. Теоретико - графових поданням мережі Петрі є двочастковий орієнтований мультіграф. Відповідно до цього граф мережі Петрі володіє двома типами вузлів:

кружок O являється позицією,

планка | являється переходом.

Орієнтовані дуги з'єднують позиції та переходи.

Приклад. Представлення мережі Петрі в виді графа і в виді структури мережі Петрі.

Нехай задана наступна структура мережі Петрі: C = (P,T,I,O),

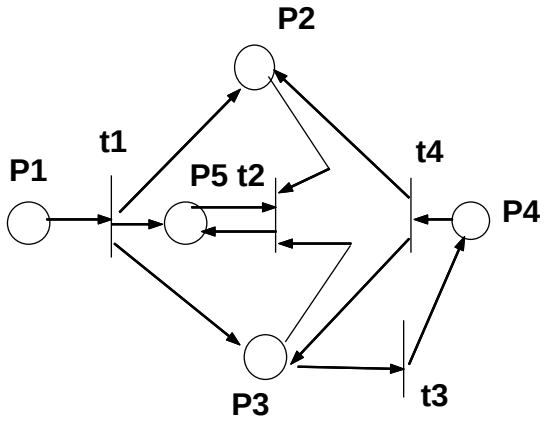


Рис. 1

n=5, m=4

P = {p1,p2,p3,p4,p5} T = {t1,t2,t3,t4}

I(t1)={p1} O(t1)={p2,p3,p5}

I(t2)={p2,p3,p5} O(t2)={p5}

I(t3)={p3} O(t3)={p4}

I(t4)={p4} O(t4)={p2,p3}

Для мережі, зображеної на рис. 1 розширеними вхідний і вихідний функціями є:

I(p1)={} O(p1)={t1}

I(p2)={t1,t4} O(p2)={t2}

I(p3)={t1,t4} O(p3)={t2,t3}

I(p4)={t3} O(p4)={t4}
I(p5)={t1,t2} O(p5)={t2}

Розглянемо задачу моделювання роботи автомата з виробництва будь-якого виробу. Автомат знаходиться в стані очікування до появи заготовки, яку він обробляє і посилає в накопичувач, тобто подіями для такої системи є:

1. заготовка надійшла;
2. автомат починає обробку;
3. автомат закінчує обробку;
4. деталь надсилається на транспортер.

Умовами для системи є:

1. автомат чекає;
2. заготовка завантажена;
3. автомат виконує обробку;
4. деталь оброблена.

У мережі Петрі умови моделюються позиціями, а події - переходами. При цьому входи переходу є передумовою відповідної події, а виходи - Післяумовами. Виконання умови представляється фішкою (маркером) в позиції, що відповідає цій умові. Запуск переходу видаляє фішки, які представляють виконання передумов і утворюють нові маркери, які представляють виконання Післяумови.

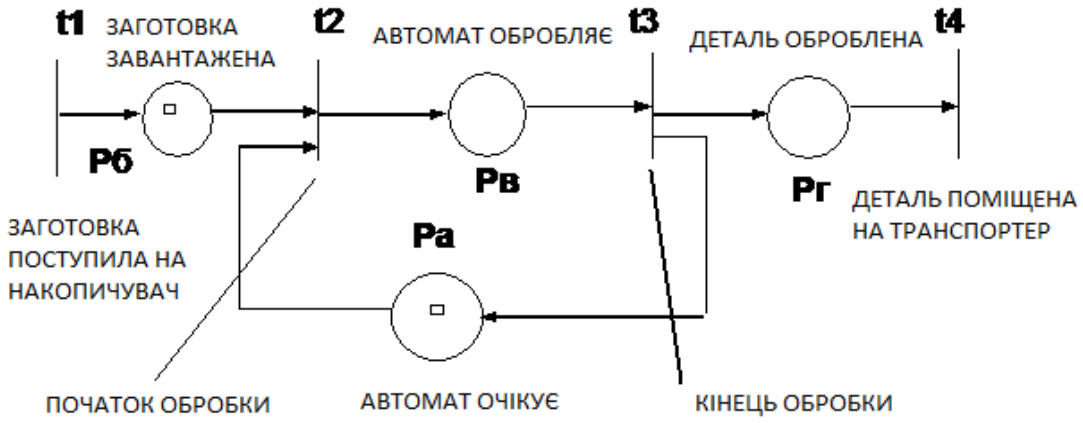


Рис.2

Аналогічний приклад можна привести для обчислювальної системи, яка обробляє завдання, що надходять з пристрою введення і виводить результати на пристрій виводу. Завдання надходить на пристрій вводу. Коли процесор вільний і в пристрої уведення є завдання, процесор починає обробку завдання. Коли завдання виконано, воно посилається на пристрій виводу; процесор або продовжує обробляти інше завдання, якщо воно є, або чекає приходу завдання. Ця система може бути промодульована мережею Петрі, зображеної на рис.3

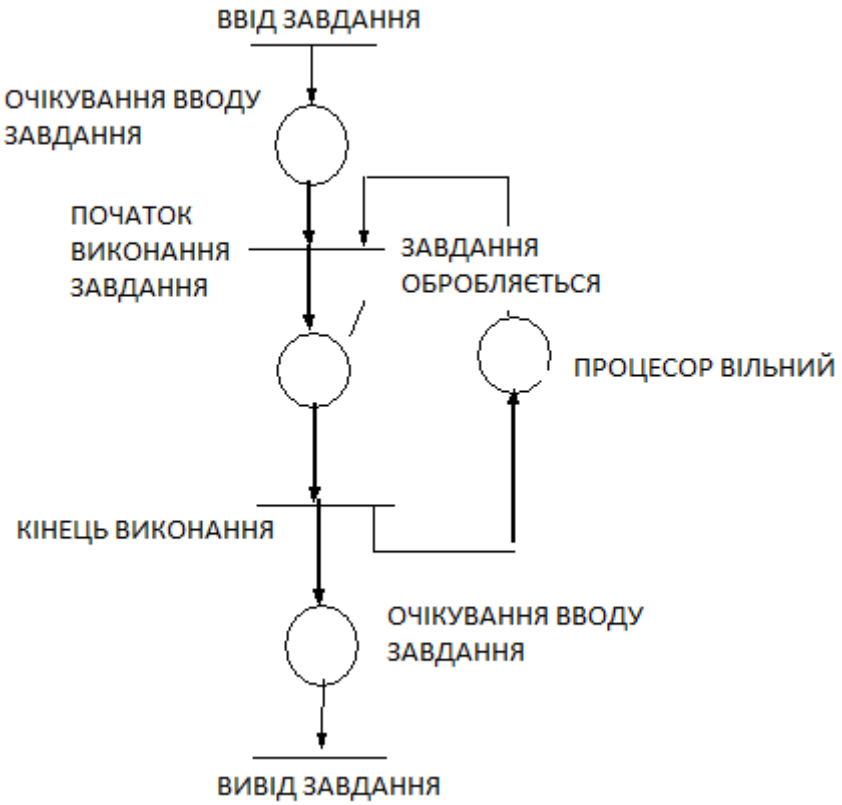


Рис.3

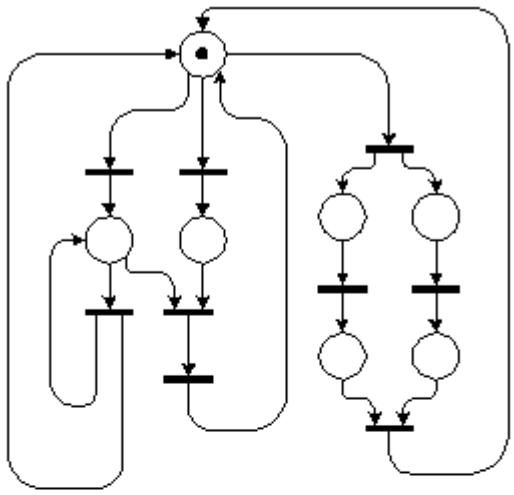
Порядок виконання роботи

Завдання 1

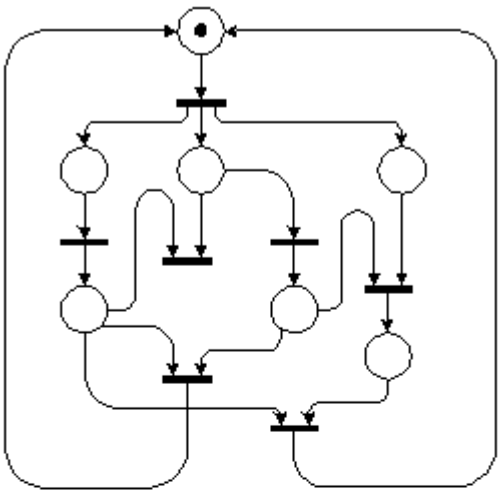
1. Вибрати структуру МП у відповідності з номером варіанту з Таблиці 1.
2. Описати задану МП-модель (P, T, I, 0, μ).

Таблиця 1. Варіанти завдань

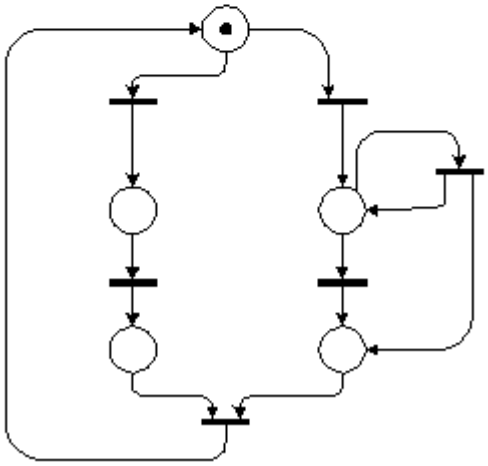
1



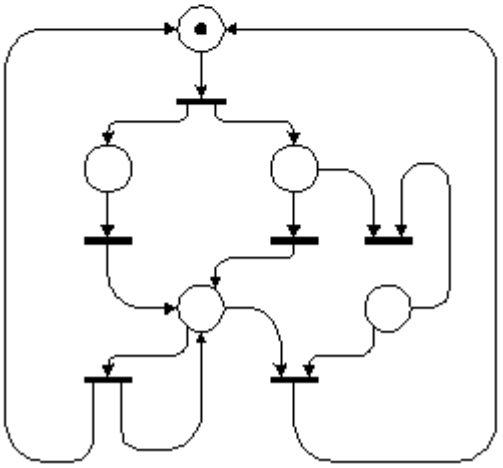
2



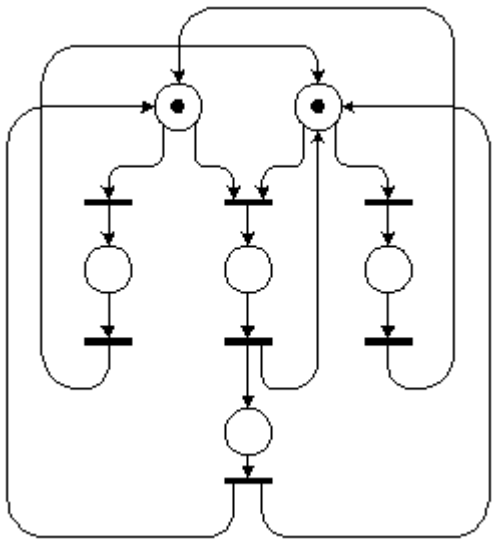
3



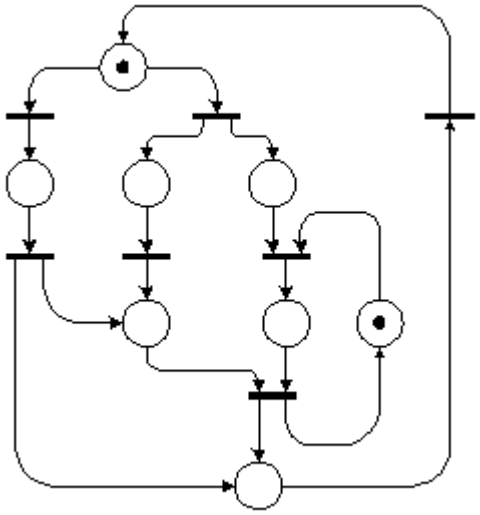
4



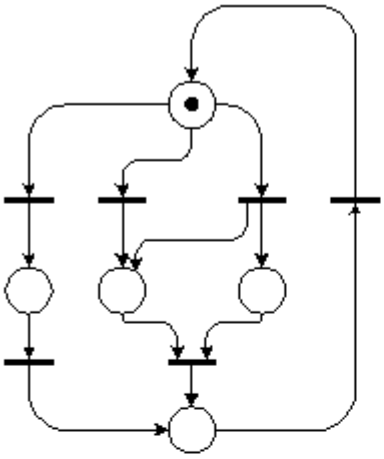
5



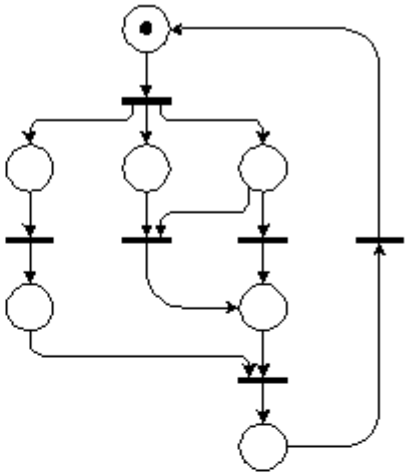
6



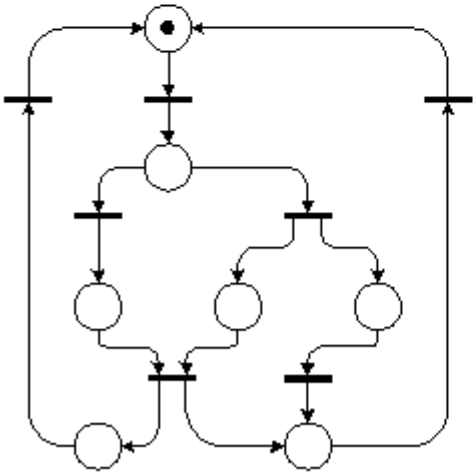
7



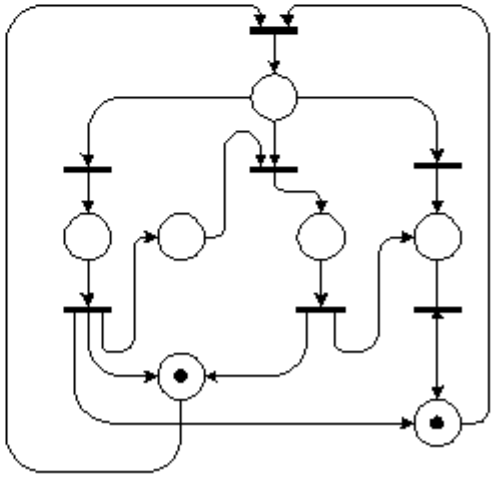
8

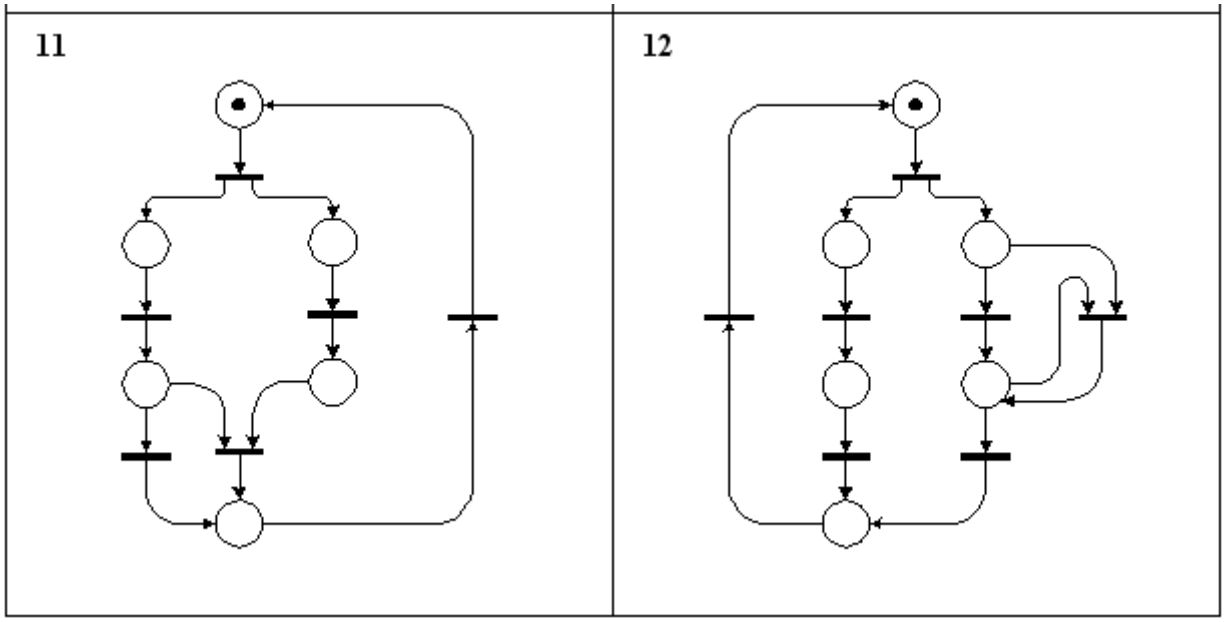


9



10





Завдання 2

1. Вибрати обчислювальну структуру відповідно з номером варіанту з Таблиці 2.
2. Розробити МП-модель відповідно до її словесного опису.
3. Провести аналіз отриманої МП-моделі
4. На основі дослідження зробити висновки про коректності моделі.

Таблиця 2. Варіанти завдань

1.	Дана обчислювальна структура, яка складається з двох незалежних підканалів ПКВ1, який вводить дані, і ПКВ2, який виводить дані. Обробка даних ведеться на конвеєрному процесорі, що складається з трьох процесорних елементів. Якщо працює процесор, то введення даних заборонений.
2.	Дана обчислювальна структура, яка включає канал введення-виведення, що складається з підканалів ПКВ1, ПКВ2, ПКВ3, і паралельний процесор, що складається з трьох процесорних елементів ПЕ1, ПЕ2, ПЕ3. Введення даних виконують підканали ПКВ1 і ПКВ2, висновок - підканал ПКВ2. Підканал ПКВ3 керує передачею даних в процесорні елементи: ПЕ1 займає підканал ПКВ3 на весь час обробки даних, ПЕ2 - тільки на час введення і виведення, ПЕ3 - тільки на час виведення.
3.	Дано обчислювальні структури ВС1 і ВС2. ВС1 має паралельний процесор, що складається з двох процесорних елементів. ВС2 має конвеєрний процесор, також складається з двох процесорних елементів. Канал вводу-виводу включає два підканали ПКВ1 і ПКВ2. Введення та обробка даних в ВС1 проводиться під управлінням підканала ПКВ1, а в ВС2 - під управлінням підканала

	ПКВ2. Висновок даних з ВС1 і ВС2 вимагає заняття каналу введення-виведення повністю.
4.	Дано обчислювальні структури ВС1 і ВС2, які мають відповідно паралельний (ПЕ1 | | ПЕ2 | | ПЕ3) і послідовний (ПЕ1-ПЕ2) процесори. Обробка даних в процесорах ВС1 і ВС2 починається одночасно. Канал вводу-виводу має один піодканал і виконує введення і виведення даних у кожній обчислювальної структурі.
5.	Дано обчислювальні структури ВС1, ВС2, ВС3 і канал введення-виведення, що складається з підканалів ПКВ1, ПКВ2, ПКВ3. ВС1 виконує введення даних з використанням підканалів ПКВ1 і ПКВ2. ВС2 виконує обробку даних на процесорі з наступною структурою ((ПЕ1 | | ПЕ2)-ПЕ3). ВС3 виконує висновок даних з використанням підканалів ПКВ2 і ПКВ3.
6.	Дано обчислювальні структури ВС1, ВС2, ВС3 і канал введення-виведення, який включає два підканали ПКВ1 і ПКВ2. ВС1 вводить дані з використанням підканалів ПКВ1 і ПКВ2. ВС2 виводить дані з використанням підканала ПКВ2. Обробка ведеться ВС3 на послідовно-паралельному процесорі зі структурою (ПЕ1 (ПЕ2 | | ПЕ3)).
7.	Дана обчислювальна структура і канал введення-виведення, який може використовуватися при введенні і виведенні даних одночасно. Обробка даних ведеться на паралельному процесорі зі структурою (ПЕ1 | | ПЕ2 | | ПЕ3).
8.	Дана конвеєрна система, яка включає обчислювальні структури ВС1, ВС2, ВС3 і канал введення-виведення з підканалів ПКВ1 і ПКВ2. ВС1 і підканал ПКВ1 вводять дані, ВС2 і підканал ПКВ2 виводять дані, ВС3 виконує обробку. Обробка ведеться на процесорі зі структурою ((ПЕ1 | | ПЕ2)-ПЕ3-(ПЕ4 | | ПЕ5)).
9.	Дана паралельна система, яка включає обчислювальні структури ВС1, ВС2, ВС3 і канал введення-виведення, який вводить і виводить дані у всі структури синхронно. Кожна обчислювальна структура має послідовний процесор, що складається з двох процесорних елементів ПЕ1 і ПЕ2. Умовою початку роботи ПЕ2 в ВС2 є закінчення обробки даних у ВС3, а умовою початку роботи ПЕ2 в ВС1 є закінчення обробки даних у ВС2.
10.	Дано обчислювальні структури ВС1, ВС2, ВС3 і ВС4. Всі обчислювальні структури обмінюються даними з одним і тим же буфером. Передача даних здійснюється каналом введення-виведення, що містить підканал ПКВ1. Процесори обчислювальних структур є послідовними і складаються з двох процесорних

	елементів. Обробку даних обчислювальні структури ведуть у такому порядку: ВС1, ВС3, ВС4, ВС2.
11.	Дано обчислювальні структури ВС1, ВС2 і ВС3, робота яких організована паралельно. Канал вводу-виводу вводить і виводить дані для всіх обчислювальних структур синхронно. Процесори обчислювальних структур мають такий вигляд: ВС1 - (ПЕ1-ПЕ2), ВС2 - (ПЕ1 | | ПЕ2), ВС3 - (ПЕ1-(ПЕ2 | | ПЕ3)). На роботу ВС1 і ВС2 накладається таке обмеження: ВС2 може почати обробку даних лише після того, як почалася обробка даних у ВС1.
12.	Дано обчислювальні структури ВС1, ВС2 і ВС3, пов'язані з кільцевою системою. ВС1 і ВС3 здійснюють введення і обробку даних, ВС2 тільки обробку. Дано один канал введення-виведення, який займається на весь час роботи обчислювальною структури. Процесори обчислювальної структури мають наступну організацію: ВС1 - (ПЕ1 | | ((ПЕ2 | | ПЕ3)-ПЕ4)), ВС2 - (ПЕ1-ПЕ2-ПЕ3), ВС3 - (ПЕ1 | | (ПЕ2-ПЕ3-ПЕ4)).

Контрольні питання:

1. Що являє собою мережа Петрі?
2. Що є вхідною та вихідною позицією переходу?
3. Який вид розмітки називається тупиковою?
4. Яка розмітка називається безпечною?
5. Які вершини є граничними?
6. Як визначається р-ланцюг?

Лабораторна робота №4

Тема: Клітинні автомати.

Мета: Ознайомитись з програмою «Життя».

Короткі теоретичні відомості

Ідея клітинних автоматів з'явилася в кінці сорокових років 20 століття. Вона була задумана і сформульована Джоном фон Нейманом і Конрадом Цусе незалежно один від одного як універсальна обчислювальна середовище для побудови, аналізу та порівняння характеристик алгоритмів.

«Клітинні автомати є дискретними динамічними системами, поведінка яких повністю визначається в термінах локальних залежностей. У значній мірі саме йде мова про великий клас безперервних динамічних систем, визначених рівняннями в приватних похідних. У цьому сенсі клітинні автомати в інформатиці є аналогом фізичного поняття «Поля». Простір представлено рівномірної сіткою, кожна клітинка або клітка яка містить кілька бітів даних. Таким чином, закони системи є локальними і всюди однаковими. «Локальний» означає, що для того, щоб дізнатися, що станеться тут хвилину по тому, досить подивитися на стан найближчого оточення: ніяка дальнодійність не допускається. «Подібність» означає, що закони скрізь одні й ті ж.

Слід зазначити, що клітинні автомати - це не просто машини, які працюють з розбитим на клітини полем. Область застосування клітинних автоматів майже безмежна: від найпростіших «хрестиків-нуликів» до штучного інтелекту. Тема клітинних автоматів дуже актуальна, тому що може призвести до розгадок багатьох питань у навколишньому світі. Розробник гри «Життя» Конуей, вважав, що наш всесвіт можна уявити клітинним автоматом, який управляє рухом елементарних частинок у відповідності з деякими правилами.

Клітинні автомати використовуються для моделювання гідродинамічних і газодинамічних течій. Рівняння гідродинаміки відповідають математичної моделі, яка описує поведінку гратчастого газу, одного з клітинних автоматів, на макрорівні. Структури, що у цих клітинних автоматах, схожі на обурення поведінки поверхні потоку рідини механічною перешкодою. Примітивні одномірні клітинні автомати можуть моделювати процес горіння різного характеру. В даний час теорія клітинних автоматів найбільш перспективно належить до питання про розробку самовідтворюючих електронних ланцюгів.

Клітинні автомати застосовуються не тільки в математиці й фізиці, а також у біології, економіці, соціології, інформатиці і т.д. За допомогою клітинних автоматів успішно вирішувалися задачі моделювання течій з вільним кордоном, поширення теплових потоків, росту і динаміки доменів, зростання дендритів, опису руху натовпу. Їх можна використовувати при складанні генетичних алгоритмів.

За своєю поведінкою клітинні автомати поділяються на чотири класи. До першого класу відносяться автомати, що приходять через певний час до сталого однорідного стану. Автомати другого класу через деякий час після пуску генерують стаціонарні або періодичні в часі структури. В автоматах третього класу після деякого часу перестає спостерігатися кореляція процесу з початковими умовами. Нарешті, поведінка автоматів четвертого класу сильно

визначається початковими умовами та з їх допомогою можна генерувати різні шаблони поведінки. Такі автомати є кандидатами на прототип клітинної обчислювальної машини. Зокрема, за допомогою специфічних клітинних конфігурацій гри «Життя», яка як раз і є автоматом четвертого типу, можна побудувати дискретні елементи цифрового комп'ютера.

Ще одне застосування клітинних автоматів в інформатиці – шифрування і стиснення даних. Клітинні автомати застосовуються при реалізації ефективної системи розпізнавання образів. Один з можливих шляхів її створення – побудова динамічної системи, аттрактором якої в її конфігураційному просторі були б типові картини-образи. Початкові умови завжди опиняться в області тяжіння однієї з картин, з плином часу система трансформує початкові параметри, привівши їх до найближчої структури - аттрактору.

Часто в різних сферах діяльності виникають завдання знаходження оптимального можливого варіанту з необмеженого числа . Точного рішення, як правило, не потрібно, але дискретний комп'ютер не здатний навіть приблизно дати оптимальний результат.

Таким чином, клітинні автомати знайшли широке застосування у багатьох сферах людської діяльності, багато завдань можливо вирішити лише за допомогою комп'ютера. Розглянемо кілька прикладів клітинних автоматів.

Гра «Життя»

Напевно, найбільш відомим можна вважати клітинний автомат під назвою гра «Життя». Створена гра «Життя» була в 1970 році Джоном Хортоном Конуей, математиком Кембриджського університету. Виникаючі в процесі гри ситуації дуже схожі на реальні процеси, що відбуваються при зародженні, розвитку і загибелі колонії живих організмів. З цієї причини «Життя» можна віднести до швидко розвиваються в наші дні категорії ігор, що імітують процеси, що відбуваються в живій природі.

Розглядається нескінченна плоска решітка квадратних осередків-клітин. Час в цій грі дискретний (t=0, 1, 2, ...). Клітка може бути живою чи мертвою.

Зміна її стану в наступний момент часу t+1 визначається станом її сусідів у момент часу t (сусідів у клітини вісім, четверо мають з нею спільні ребра, а четверо тільки вершини). Основна ідея полягає в тому, щоб, почавши з якогось розташування клітин, простежити за еволюцією вихідної позиції під дією «генетичних законів» Конуея, які керують народженням, загибеллю і виживанням клітин. Конуей ретельно підбирав свої правила і довго перевіряв їх на «практиці», домагаючись, щоб вони по можливості задовільняли три умови:

• Не повинно бути ні однієї вихідної конфігурації, для якої існував би простеий доказ можливості необмеженого росту популяції.

• У той же час повинні існувати такі початкові конфігурації, які мають здатність безмежно розвиватися.

• Повинні існувати прості початкові конфігурації, які на протязі значного проміжку часу ростуть, змінюються і закінчують свою еволюцію одним з наступних трьох способів: повністю зникають (або із-за перенаселення, тобто дуже великої щільності живих клітин, або навпаки, через розрідженості клітин, що утворюють конфігурацію); переходять у стійку конфігурацію і перестають

змінюватися взагалі або виходять на коливальний режим, тобто нескінченний цикл перетворень з певним періодом.

Внаслідок цих вимог з'явилися наступні правила гри «Життя»:

1. Виживання. Кожна клітина, що має дві або три сусідні живі клітини, виживає і переходить у наступне покоління.

2. Загибель. Кожна клітина, у якої більше трьох сусідів, гине через перенаселеності. Кожна клітина, навколо якої вільні всі сусідні клітини або ж зайнята лише одна клітина, гине від самотності.

3. Народження. Якщо число зайнятих клітин, з якими межує якась порожня клітина, в точності дорівнює трьом, то на цій клітці відбувається народження нового організму.

Задамося питаннями: Які основні типи структур (тобто конфігурацій, визначають поведінку співтовариств на великих проміжку часу) можуть існувати в такій системі? Які тут закони організації структур? Чи можуть вони взаємодіяти, і до чого це призводить? З'ясуємо, які закономірності є наслідками представлених вище правил.

Перша закономірність - властивість локалізації - структури, які розділені двома «мертвими» (порожніми) клітинами не впливають одна на одну.

Друга закономірність - система клітин, яку описує гра «Життя», розвивається безповоротно. Насправді конфігурація в момент часу t повністю визначає майбутнє (стан в моменти t+1, t+2 і так далі). Але відновити минуле системи по її теперішньому не вдається. Картина тут така ж, як у одновимірних відображеннях, тільки перетворень у даної конфігурації може бути нескінченно багато. (Доведемо це твердження: скористаємося властивістю локалізації і розташуємо навколо даної конфігурації безліч локалізованих одиночних клітин або їх пар так, щоб вони не впливали на неї й одні на одних. Зрозуміло, що всі вони зникнуть на наступному кроці, жодним чином не вплинувши на майбутнє системи). Тут ми можемо помітити ознаки нелінійних дисипативних структур: ці структури визначали поведінку системи при t напрямленому до нескінченності в випадку різних початкових даних.

Третя закономірність - як показують тривалі спостереження за процесом розвитку колоній, конфігурації, що не володіли на початку гри симетрією, виявляють тенденцію до переходу в симетричні форми. Знайдені властивості симетрії в процесі подальшої еволюції не втрачаються, симетрія конфігурації може лише збагачуватися.

Домовимося класифікувати конфігурації клітин за наступними параметрами:

• За кількістю клітин в комбінації: одинична клітина, дуплет (2 клітини в комбінації), триплет (3 клітини) і т.д.

• За перспективою розвитку: розвиваючі (необмежений ріст), стабільні (кількість клітин в популяції коливається біля якогось середнього значення), вимираючі (популяція стабільно зменшується), періодичні (кількість клітин приймає декілька фіксованих значень через певний період).

Тепер розглянемо типові структури, що з'являються в грі «Життя».

Найпростішими є стаціонарні, тобто не залежать від часу структури (Сам Конуей назвав їх «любителями спокійного життя»). Їх приклади показані на

рисунку 1. За допомогою цих стаціонарних структур можна отримати безліч інших. Справді, якщо ми маємо таку структуру, то конфігурація,отримана поворотом на 90^{O},також буде стаціонарною. Зміни в нижніх рядах показують, як можна добудовувати певні структури до будь-яких розмірів.

Використовуючи властивість локалізації можна будувати «великі» стаціонарні структури з «малих»-елементарних.

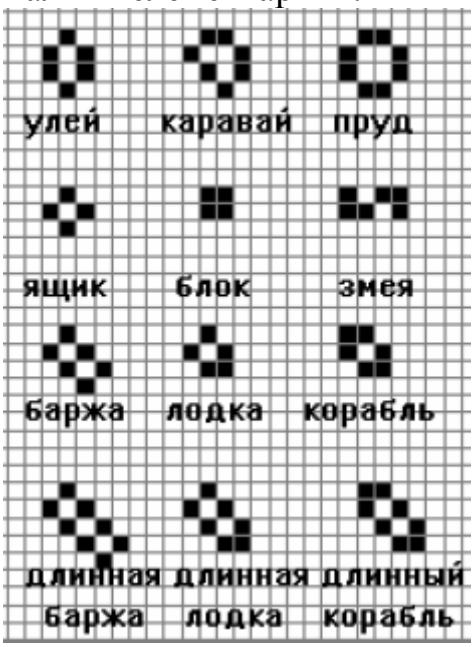


Рисунок 1. Приклади стаціонарних структур, що реалізуються у грі «Життя»

Можна вважати, що стаціонарні структури повторюють себе на кожному кроці за часом. Але є й інші конфігурації, що повторюють себе через N кроків, так звані N-цикли (періодичні структури).

Наведені приклади показують, що в обговорюваній дискретній системі існує велика кількість різних типів впорядкованості, які визначають асимптотичну поведінку деякої кількості конфігурацій (у цьому сенсі вони виявляються еквівалентні атрактори динамічних систем). Однак можна довести більше - у грі «Життя» існують як завгодно складні типи впорядкованості, ця дискретна система виявляється еквівалентною універсальною обчислювальною машиною.

ЕОМ можна розглядати як кінцевий набір найпростіших логічних елементів, що здійснюють операції І, АБО, НЕ, певним чином з'єднаних проводами, за якими поширюється набір імпульсів, кодують послідовність нулів і одиниць. Як генератор таких імпульсів у грі «Життя» виступає планерна рушниця. Наявність планера в потоці природно інтерпретує як одиницю, відсутність як нуль. Зіткнення планерів, що призводять до їх анігіляції, дозволяє побудувати елемент НЕ, направивши два потоки під прямим кутом (якщо планер у певному місці є в першому потоці, то після зіткнення планер в іншому потоці на цьому місці зникне). Більш складним чином конструюються інші елементи.

Для аналізу ситуацій, що виникають у грі «Життя», застосовується комп'ютер. У програмі, що моделює цей клітинний автомат, використовується квадратна матриця, яка і є полем для гри. При зміні ходу аналізується кожен

елемент старої матриці і будується на її основі нова, яка відповідає конфігурації на наступному кроці еволюції. Для більш детального дослідження гра Конуея розширена на декілька популяцій, кожна з яких розвивається за своїми правилами. Правила для кожної популяції вибираються з наступних:

• Умови народження і смерті. Задаються чотири параметри (параметри можна змінювати в процесі гри): мінімальна та максимальна кількість сусідів своєї популяції, при якому народжується клітина; мінімальна та максимальна кількість сусідів, при якому клітина виживає і переходить в наступне покоління.

• Сусідами клітини можуть бути будь-які клітки, що знаходяться в квадраті 3x3 з центром в даній клітині.

Гра «Життя» знайшла своє застосування в біології.

Завдання: Гра «Життя»

1) Стандартні правила

• Дослідити правила еволюції в грі. Розглянути динаміку найпростіших популяцій: одна клітина, дві клітини, різні конфігурації з трьох і чотирьох клітин. Знайти п'ять триплетів (конфігурацій, що складаються з трьох клітин), які не гинуть на наступному кроці еволюції.

• Розглянути динаміку всіх вищевказаних конфігурацій.

• Розглянути динаміку конфігурацій на рис. 2, визначити характер їх поведінки. У випадку періодичної динаміки знайти період коливань.

• Знайти свої власні початкові конфігурації у грі «Життя», які б задовольняли наступні вимоги:

а) з плином часу популяція «гине».

б) з плином часу в системі спостерігається становлення стаціонарного стану

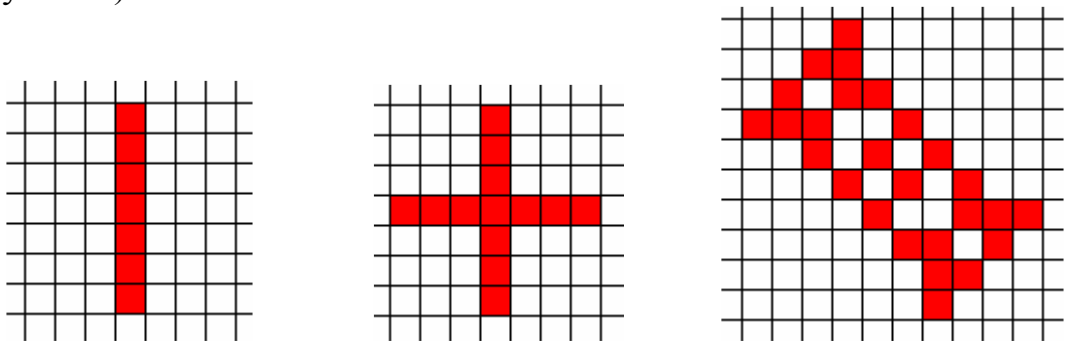
в) після перехідного процесу в системі спостерігаються періодичні коливання з періодом 2.

г) після перехідного процесу в системі спостерігаються періодичні коливання з періодом 3.

д) після перехідного процесу в системі спостерігаються періодичні коливання з періодом 4.

е) популяція за досить великий час (більше 70 кроків) не приходить до якої-небуть регулярної поведінки чи стійкого стану.

• Дослідити динаміку конфігурації під назвою Глайдерна рушниця (Рисунок 3)



а б в

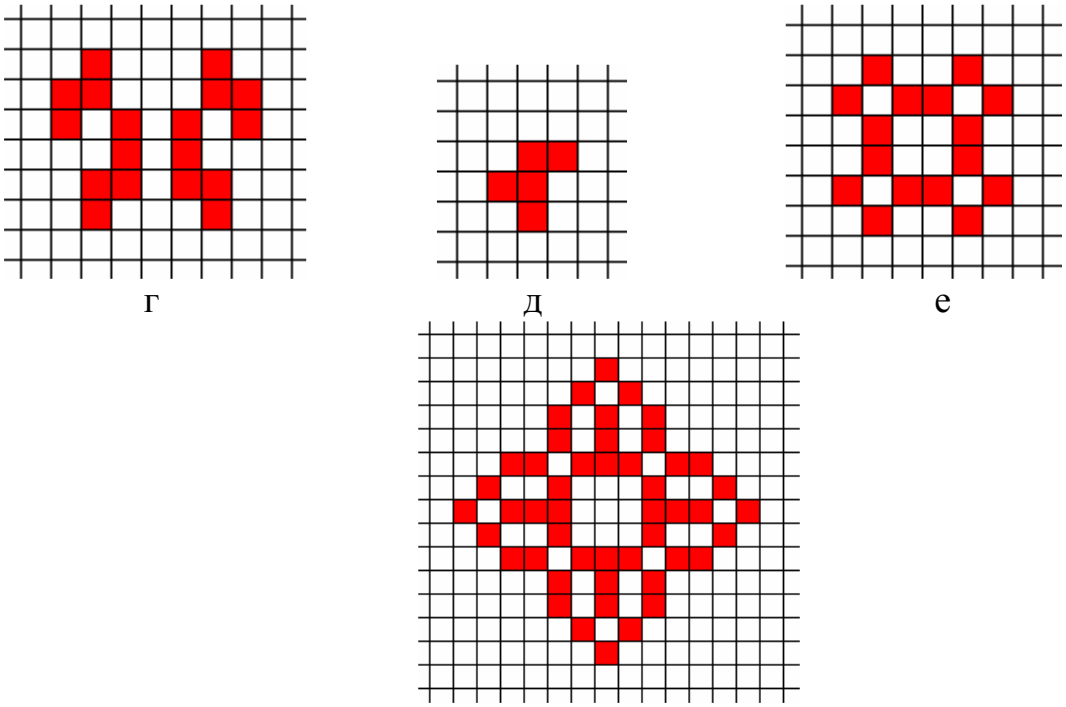


Рисунок 2

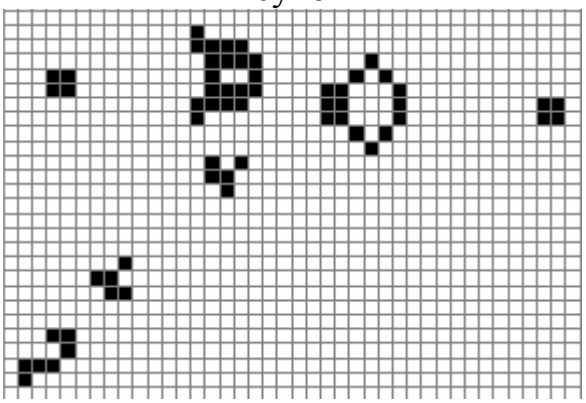


Рисунок 3. Планерна рушниця (катапульта) - конфігурація, яка генерується за кожні 30 ходів планер.

Контрольні питання:

1. Що таке клітинний автомат?
2. Застосування клітинних автоматів?
3. Охарактеризуйте структуру гри «Життя».
4. Як відбувається народження нових клітин?
5. Які закономірності є в грі «Життя»?
6. Як проводиться класифікація клітин?

Лабораторна робота № 5

Тема. Моделювання систем за допомогою засобів Matlab Simulink

Мета. Навчитися моделювати за допомогою засобів Simulink

Короткі теоретичні відомості

1. Загальні відомості

Програма **Simulink** є додатком до пакету **MATLAB.** При моделюванні з використанням **Simulink** реалізується принцип візуального програмування, відповідно до якого, користувач на екрані з бібліотеки стандартних блоків створює модель пристрою і здійснює розрахунки. При цьому, на відміну від класичних способів моделювання, користувачеві не потрібно досконально вивчати мову програмування і чисельні методи математики, а досить загальних знань потрібних при роботі на комп'ютері і, природно, знань тієї предметної області в якій він працює.

Simulink є досить самостійним інструментом **MATLAB** і при роботі з ним зовсім не потрібно знати сам **MATLAB** і інші його додатки. З іншого боку доступ до функцій **MATLAB** і іншим його інструментам залишається відкритим і їх можна використовувати в **Simulink.** Частина входять до складу пакетів має інструменти, що вбудовуються в **Simulink** (наприклад, **LTI-Viewer** додатки **Control System Toolbox** - пакета для розробки систем управління). Є також додаткові бібліотеки блоків для різних областей застосування (наприклад, **Power System Blockset** - моделювання електротехнічних пристроїв, **Digital Signal Processing Blockset** - набір блоків для розробки цифрових пристроїв і т.д).

При роботі з **Simulink** користувач має можливість модернізувати бібліотечні блоки, створювати свої власні, а також складати нові бібліотеки блоків.

При моделюванні користувач може вибирати метод розв'язання диференціальних рівнянь, а також спосіб зміни модельного часу (з фіксованим або змінним кроком). У ході моделювання є можливість стежити за процесами, що відбуваються в системі. Для цього використовуються спеціальні пристрої спостереження, входять до складу бібліотеки **Simulink.** Результати моделювання можуть бути представлені у вигляді графіків або таблиць.

Перевага **Simulink** полягає також у тому, що він дозволяє поповнювати бібліотеки блоків за допомогою підпрограм написаних як мовою **MATLAB,** так і на мовах **C + +, Fortran** і **Ada.**

2. Запуск Simulink

Для запуску програми необхідно попередньо запустити пакет **MATLAB.** Основне вікно пакета **MATLAB** показано на Рис. 1. Там же показана підказка з'являється у вікні при наведенні покажчика миші на ярлик **Simulink** в панелі інструментів.

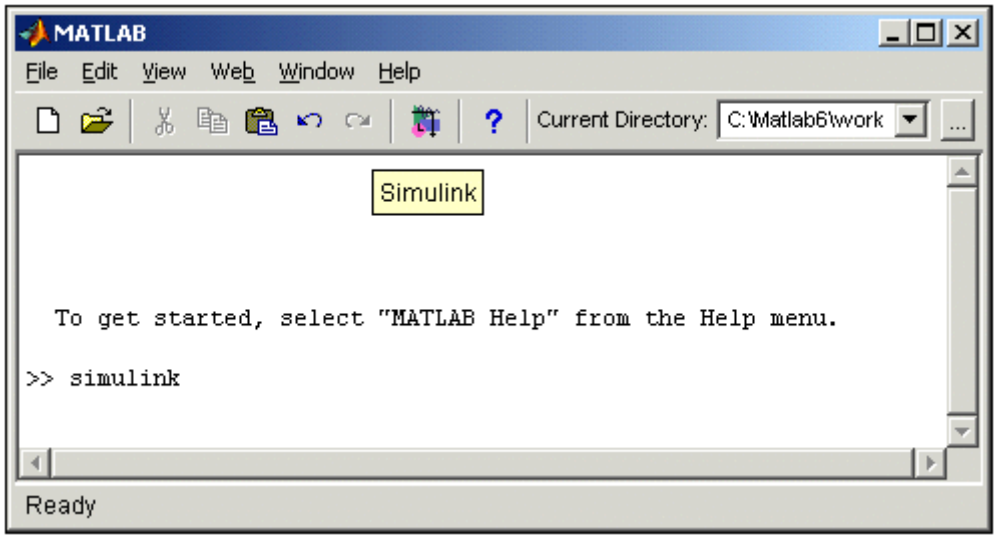


Рис 1. Основне вікно програми **MATLAB**

Після відкриття основного вікна програми **MATLAB** потрібно запустити програму **Simulink.** Це можна зробити одним з трьох способів:

- Натиснути кнопку **(Simulink)** на панелі інструментів командного вікна **MATLAB.**
- У командному рядку головного вікна **MATLAB** надрукувати **Simulink** і натиснути клавішу **Enter** на клавіатурі.
- Виконати команду **Open ...** в меню **File** і відкрити файл моделі **(mdl** - файл).

Останній варіант зручно використовувати для запуску вже готової і налагодженої моделі, коли потрібно лише провести розрахунки і не потрібно додавати нові блоки в модель. Використання першого і другого способів призводить до відкриття вікна оглядача розділів бібліотеки **Simulink** (рис. 2).

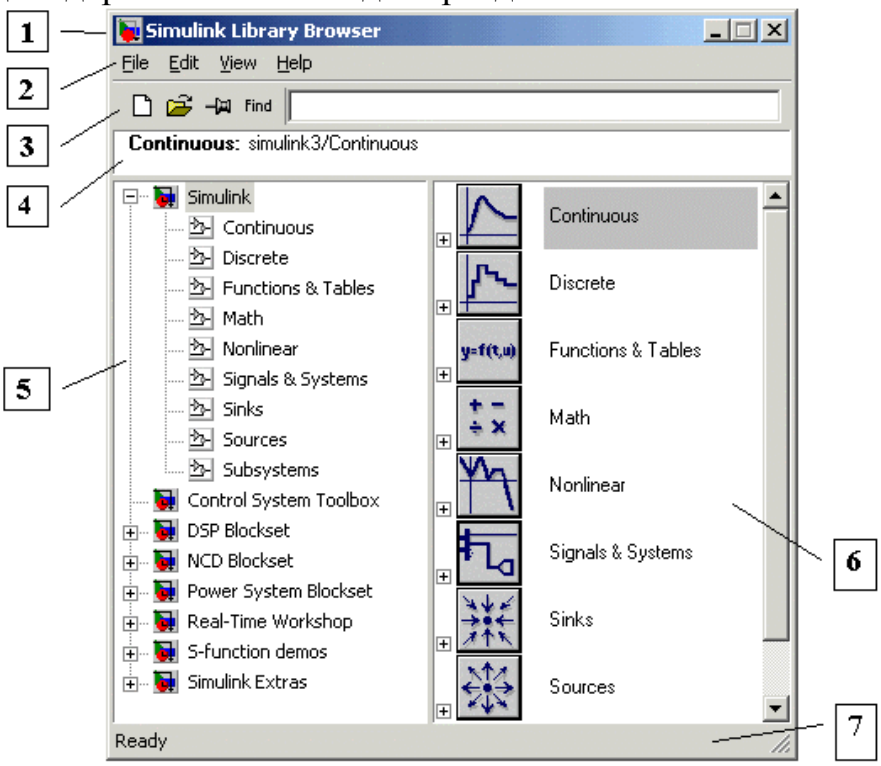


Рис 2. Вікно оглядача розділів бібліотеки **Simulink**

3. Оглядач розділів бібліотеки Simulink

Вікно оглядача бібліотеки блоків містить такі елементи (Рис.2):

1. Тема, з назвою вікна - **Simulink Library Browser.**
2. Меню, з командами **File, Edit, View, Help.**
3. Панель інструментів, з ярликами найбільш часто використовуваних команд.
4. Вікно коментаря для виведення поясняющего повідомлення про обраний блоці.
5. Список розділів бібліотеки, реалізований у вигляді дерева.
6. Вікно вмісту розділу бібліотеки (список вкладених розділів бібліотеки або блоків)
7. Рядок стану, що містить підказку по виконуваному дії.

На рис. 2 виділена основна бібліотека **Simulink** (у лівій частині вікна) і показані її розділи (у правій частині вікна).

Бібліотека **Simulink** містить наступні основні розділи:

1. **Continuous -** лінійні блоки.
2. **Discrete -** дискретні блоки.
3. **Functions & Tables -** функції та таблиці.
4. **Math** - блоки математичних операцій.
5. **Nonlinear -** нелінійні **блоки.**
6. **Signals & Systems** - сигнали і системи.
7. **Sinks** - реєструючі пристрої.
8. **Sources** - джерела сигналів і впливів.
9. **Subsystems -** блоки підсистем.

Список розділів бібліотеки **Simulink** представлений у вигляді дерева, і правила роботи з ним є загальними для списків такого виду:

- Піктограма згорнутого вузла дерева містить символ **"+",** а піктограма розгорнутого містить символ **"-".**
- Для того щоб розгорнути або згорнути вузол дерева, досить клацнути на його піктограмі лівою клавішею миші (ЛКМ).

При виборі відповідного розділу бібліотеки в правій частині вікна відображається його вміст (Рис. 3).

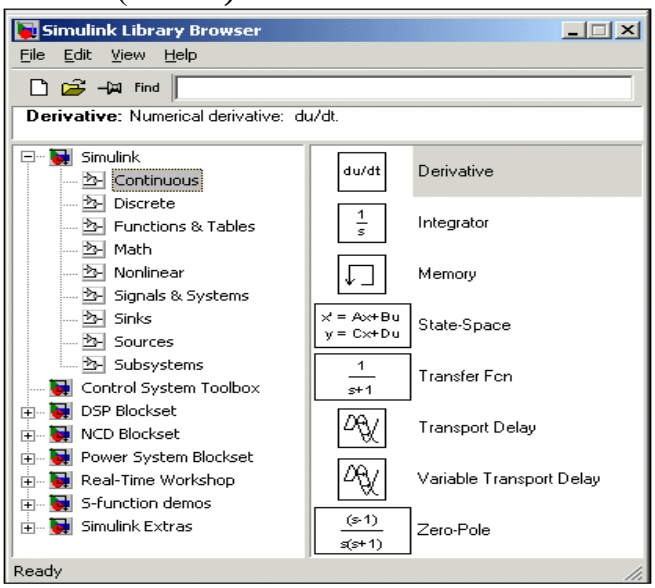


Рис 3.Вікно оглядача з набором блоків розділу бібліотеки

Для роботи з вікном використовуються команди зібрані в меню. Меню оглядача бібліотек містить наступні пункти:

- **File (Файл)** - Робота з файлами бібліотек.
- **Edit (Редагування)** - Додавання блоків і їх пошук (по назві).
- **View (Вид)** - Управління показом елементів інтерфейсу.
- **Help (Довідка)** - Висновок вікна довідки за оглядачеві бібліотек.

Рис 4. Панель інструментів оглядача розділів бібліотек

Кнопки панелі інструментів мають таке призначення:

1. Створити нову **S-модель** (відкрити нове вікно моделі).
2. Відкрити одну з існуючих **S-моделей.**
3. Змінити властивості вікна оглядача. Дана кнопка дозволяє встановити режим відображення вікна оглядача *"поверх всіх вікон"*. Повторне натискання скасовує такий режим.
4. Пошук блоку за назвою (по перших символах назви). Після того як блок буде знайдений, у вікні оглядача відкриється відповідний розділ бібліотеки, а блок буде виділено. Якщо ж блок з такою назвою відсутня, то у вікні коментаря буде виведено повідомлення **Not found <ім'я блоку>** (Блок не знайдене).

4. Створення моделі

Для створення моделі в середовищі **SIMULINK** необхідно послідовно виконати ряд дій:

4.1. Створити новий файл моделі за допомогою команди **File / New / Model,** або використовуючи кнопку на панелі інструментів (тут і далі, за допомогою символу "/", зазначені пункти меню програми, які необхідно послідовно вибрати для виконання зазначеної дії). Новостворене вікно моделі показано на Рис. 5.

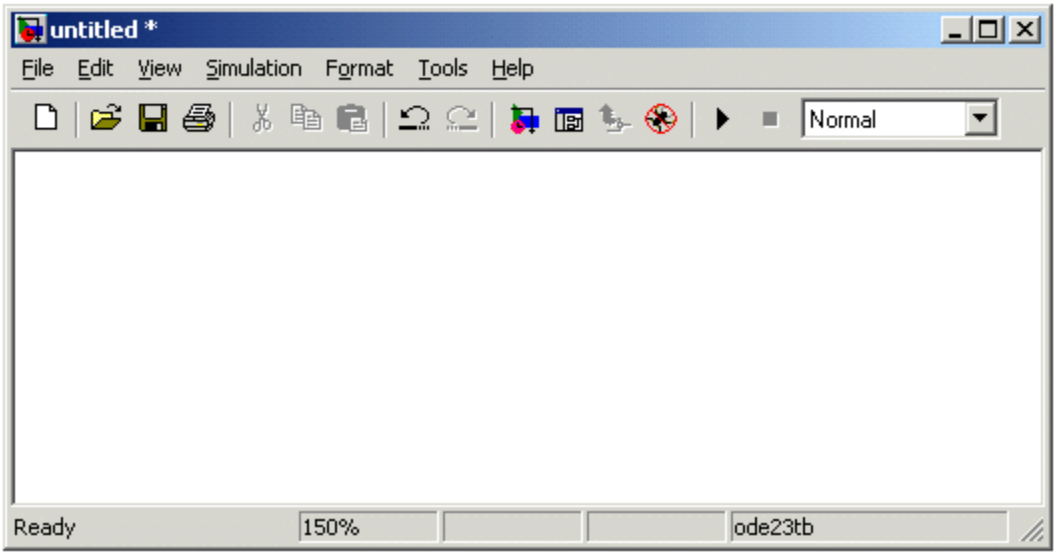


Рис 5. Порожнє вікно моделі

4.2. Розташувати блоки у вікні моделі. Для цього необхідно відкрити відповідний розділ бібліотеки (Наприклад, **Sources - Джерела).** Далі, вказавши курсором на необхідний блок і натиснувши на ліву клавішу "миші" -

"перетягнути" блок в створене вікно. ***Клавішу миші потрібно тримати натиснутою.*** На Рис 6 показано вікно моделі, що містить блоки.

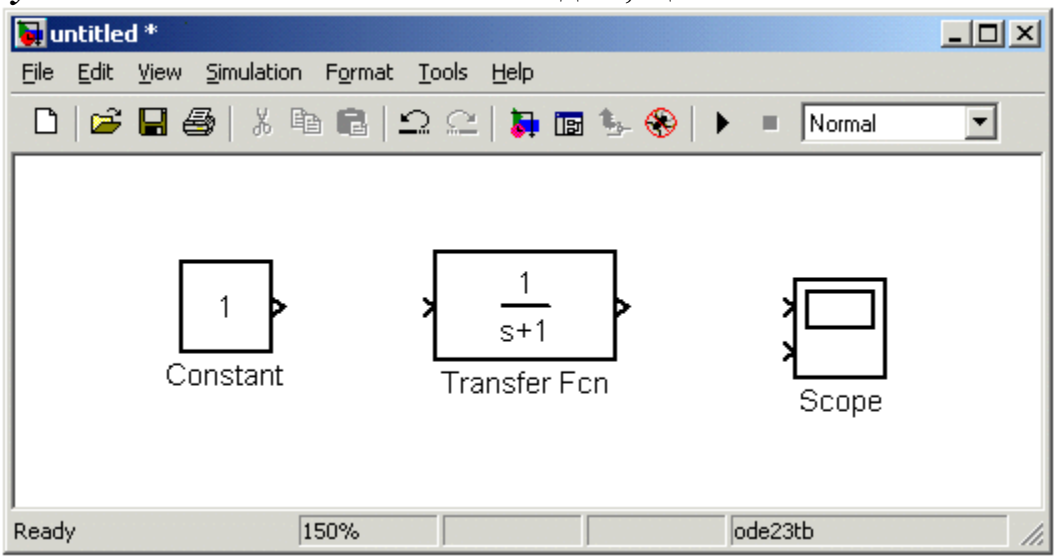


Рис 6. Вікно моделі, що містить блоки

Для видалення блоку необхідно вибрати блок (вказати курсором на його зображення і натиснути ліву клавішу "миші"), а потім натиснути клавішу **Delete** на клавіатурі.

Для зміни розмірів блоку потрібно вибрати блок, встановити курсор в один з кутів блоку і, натиснувши ліву клавішу "миші", змінити розмір блоку (курсор при цьому перетвориться в двосторонню стрілку).

4.3. Далі, якщо це потрібно, потрібно змінити параметри блоку, встановлені програмою "за замовчуванням". Для цього необхідно двічі клацнути лівою клавішею "миші", вказавши курсором на зображення блоку. Відкриється вікно редагування параметрів даного блоку. При завданні чисельних параметрів слід мати на увазі, що як десяткового роздільника повинна використовуватися точка, а не кома. Після внесення змін потрібно закрити вікно кнопкою **OK.** На рис.4.3 як приклад показані блок, що моделює передавальну функцію і вікно редагування параметрів даного блоку.

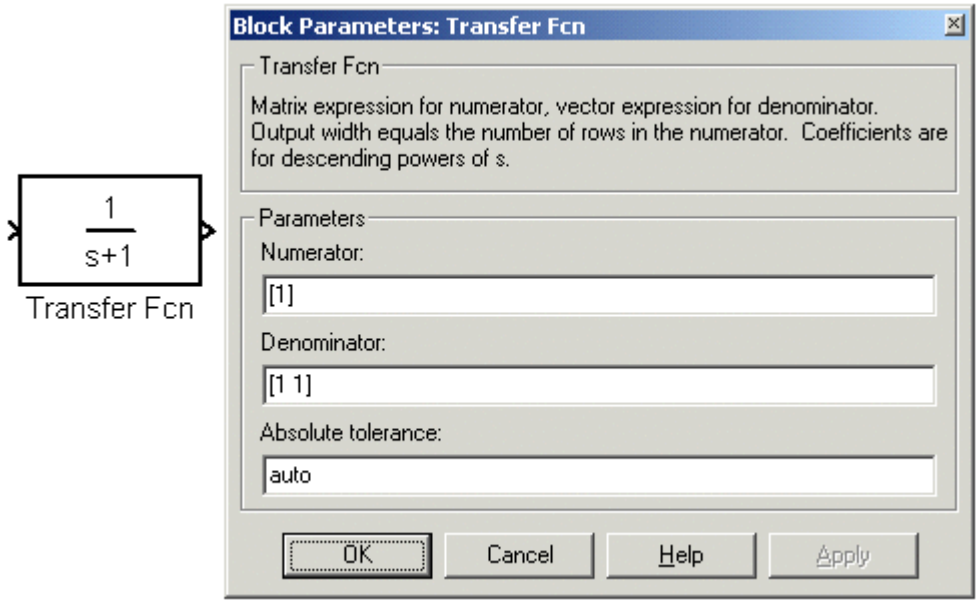


Рис 4.3. Блок, що моделює передавальну функцію і вікно редагування параметрів блоку

4.4. Після установки на схемі всіх блоків із потрібних бібліотек потрібно виконати з'єднання елементів схеми. Для з'єднання блоків необхідно вказати курсором на "вихід" блоку, а потім, натиснути і, не відпускаючи ліву клавішу "миші", провести лінію до входу іншого блоку. Після чого відпустити клавішу. У разі правильного з'єднання зображення стрілки на вході блоку змінює колір. Для створення точки розгалуження в сполучній лінії потрібно підвести курсор до передбачуваного вузла і, натиснувши *праву* клавішу "миші", протягнути лінію. Для видалення лінії потрібно вибрати лінію (так само, як це виконується для блоку), а потім натиснути клавішу **Delete** на клавіатурі. Схема моделі, в якій виконані з'єднання між блоками, показана на Рис. 4.4 .

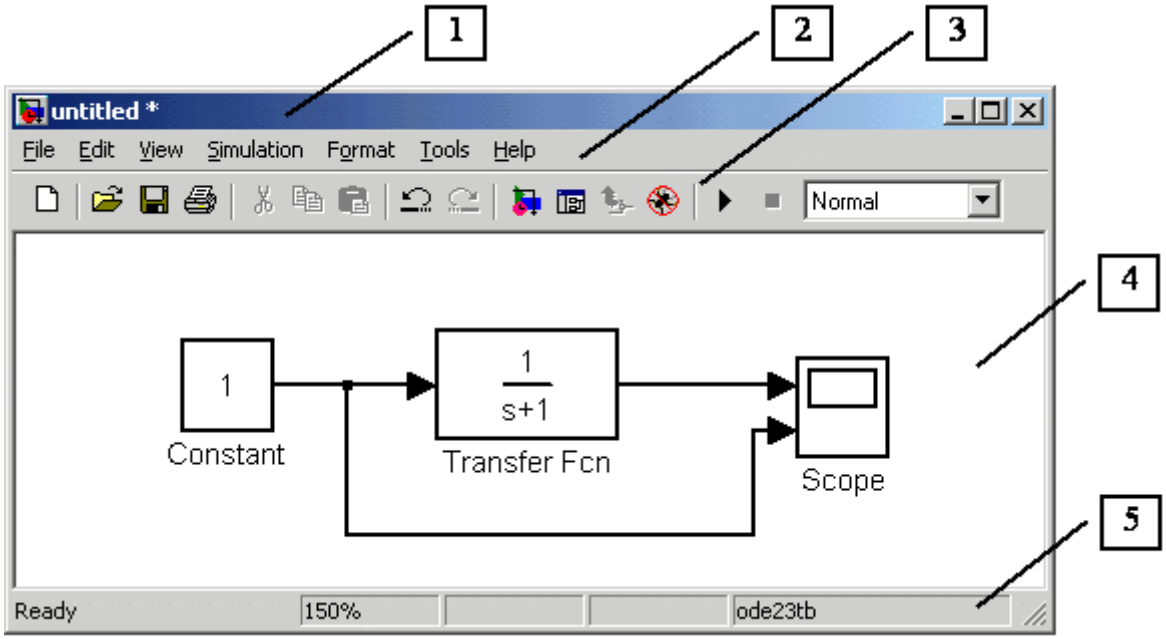


Рис 4.4. Схема моделі

4.5. Після складання розрахункової схеми необхідно зберегти її у вигляді файлу на диску, вибравши пункт меню **File / Save As ...** у вікні схеми і вказавши папку та ім'я файлу. Слід мати на увазі, що ім'я файлу не повинен перевищувати 32 символів, повинно починатися з літери і не може містити символи кирилиці і спецсимволи. Ця ж вимога стосується і до шляху файлу (до тих папках, в яких зберігається файл). При подальшому редагуванні схеми можна користуватися пунктом меню **Fille / Save.** При повторних запусках програми **SIMULINK** завантаження схеми здійснюється за допомогою меню **File / Open ...** у вікні оглядача бібліотеки або з основного вікна **MATLAB.**

5. Вікно моделі

Вікно моделі містить такі елементи (див. рис. 4.4):

1. Тема, з назвою вікна. Новоствореному вікна присвоюється ім'я **Untitled** з відповідним номером.
2. Меню з командами **File, Edit, View** і т.д.
3. Панель інструментів.
4. Вікно для створення схеми моделі.
5. Рядок стану, що містить інформацію про поточний стан моделі.

Меню вікна містить команди для редагування моделі, її налаштування і управління процесом розрахунку, роботи файлами тощо:

- File **(Файл)** - Робота з файлами моделей.

- **Edit (Редагування)** - Зміна моделі та пошук блоків.
- **View (Вид)** - Управління показом елементів інтерфейсу.
- **Simulation (Моделювання)** - Завдання налаштувань для моделювання і керування процесом розрахунку.
- **Format (Форматування)** - Зміна зовнішнього вигляду блоків і моделі в цілому.
- **Tools (Інструментальні засоби)** - Застосування спеціальних засобів для роботи з моделлю (відладчик, лінійний аналіз і т.п.)
- **Help (Довідка)** - Висновок вікон довідкової системи.

Для роботи з моделлю можна також використовувати кнопки на панелі інструментів (Рис.5.1).

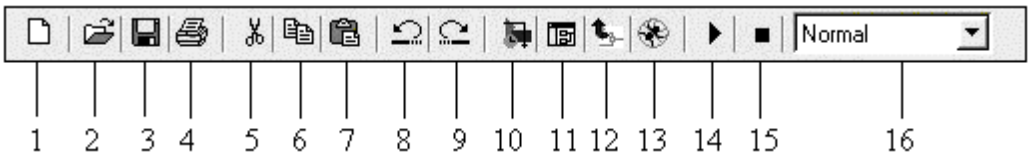


Рис 5.1. Панель інструментів вікна моделі

Кнопки панелі інструментів мають таке призначення:

1. **New Model** - Відкрити нове (пусте) вікно моделі.
2. **Open Model** - Відкрити існуючий **mdl-файл.**
3. **Save Model** - Зберегти **mdl-файл** на диску.
4. **Print Model** - Виведення на друк блок-діаграми моделі.
5. **Cut** - Вирізати виділену частину моделі в буфер проміжного зберігання.
6. **Copy** - Скопіювати виділену частину моделі в буфер проміжного зберігання.
7. **Paste** - Вставити у вікно моделі вміст буфера проміжного зберігання.
8. **Undo** - Скасувати попередню операцію редагування.
9. **Redo** - Відновити результат скасованої операції редагування.
10. **Library Browser** - Відкрити вікно оглядача бібліотек.
11. **Toggle Model Browser** - Відкрити вікно оглядача моделі.
12. **Go to parent system** - Перехід з підсистеми в систему вищого рівня ієрархії ("родітельсую систему"). Команда доступна тільки, якщо відкрита підсистема.
13. **Debug** - Запуск відладчика моделі.
14. **Start / Pause / Continue Simulation** - Запуск моделі на виконання (команда **Start);** після запуску моделі на зображенні кнопки виводиться символ ⏸, І їй відповідає вже команда **Pause** (Призупинити моделювання); для поновлення моделювання слід клацнути по тій же кнопці, оскільки в режимі паузи їй відповідає команда **Continue** (Продовжити).
15. **Stop** - Закінчити моделювання. Кнопка стає доступною після початку моделювання, а також після виконання команди **Pause.**
16. **Normal / Accelerator** - Звичайний / Прискорений режим розрахунку. Інструмент доступний, якщо встановлено додаток **Simulink Performance Tool.**

У нижній частині вікна моделі знаходиться рядок стану, в якій відображаються короткі коментарі до кнопок панелі інструментів, а також до пунктів меню, коли покажчик миші знаходиться над відповідним елементом інтерфейсу. Це ж текстове поле використовується і для індикації стану **Simulink: Ready** (Готовий) або **Running** (Виконання). У рядку стану відображаються також:

- масштаб відображення блок-діаграми (у відсотках, вихідне значення дорівнює 100%),
- індикатор ступеня завершеності сеансу моделювання (з'являється після запуску моделі),
- поточне значення модельного часу (виводиться також тільки після запуску моделі),
- використовуваний алгоритм розрахунку станів моделі (метод рішення).

6. Основні прийоми підготовки і редагування моделі

6.1. Додавання текстових написів

Для підвищення наочності моделі зручно використовувати текстові написи. Для створення напису потрібно вказати мишею місце написи і двічі клацнути лівою клавішею миші. Після цього з'явиться прямокутна рамка з курсором вводу. Аналогічним чином можна змінити і підписи до блоками моделей .. На рис. 6.1 показані текстова напис і зміна написи в блоці передавальної функції. Слід мати на увазі, що розглянута версія програми **(Simulink 4)** не адаптована до використання кириличних шрифтів, і застосування їх може мати самі різні наслідки: - відображення написів у нечитабельним вигляді, обрізання написів, повідомлення про помилки, а також неможливість відкрити модель після її збереження. Тому, застосування написів російською мовою для поточної версії **Simulink** вкрай не бажано.

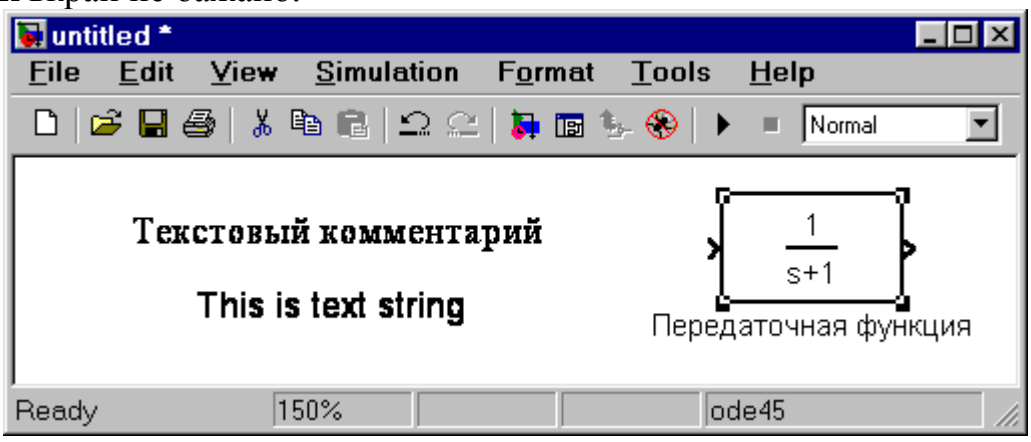


Рис 6.1. Текстова напис і зміна написи в **Transfer Function**

6.2. Виділення об'єктів

Для виконання будь-якої дії з елементом моделі (блоком, сполучною лінією, написом) цей елемент необхідно спочатку виділити.

Виділення об'єктів найпростіше здійснюється мишею. Для цього необхідно встановити курсор миші на потрібному об'єкті і клацнути лівою клавішею миші. Відбудеться виділення об'єкта. Про це будуть свідчити маркери по кутах об'єкта (див. рис. 6.1). Можна також виділити кілька об'єктів. Для цього треба встановити курсор миші поблизу групи об'єктів, натиснути ліву кнопку миші і, не відпускаючи її, почати переміщати мишу. З'явиться пунктирна рамка,

розміри якої будуть змінюватися при переміщенні миші. Усі охоплені рамкою об'єкти стають виділеними. Виділити всі об'єкти також можна, використовуючи команду **Edit / Select All.** Після виділення об'єкта його можна копіювати або переміщати в буфер проміжного зберігання, витягати з буфера, а також видаляти, використовуючи стандартні прийоми роботи в **Windows-програмах.**

6.3. Копіювання і переміщення об'єктів в буфер проміжного зберігання

Для копіювання об'єкта в буфер його необхідно попередньо виділити, а потім виконати команду **Edit / Copy** або скористатися інструментом  на панелі інструментів.

Для вирізання об'єкта в буфер його необхідно попередньо виділити, а потім виконати команду **Edit / Cut** або скористатися інструментом  на панелі інструментів. При виконанні даних операцій слід мати на увазі, що об'єкти поміщаються у власний буфер **MATLAB** і недоступні з інших додатків. Використання команди **Edit / Copy model to Clipboard** дозволяє помістити *графічне зображення* моделі в буфер **Windows** і, відповідно, робить його доступним для решти програм.

Копіювання можна виконати і таким чином: натиснути *праву* ю клавішу миші, і не відпускаючи її, перемістити об'єкт. При цьому буде створено копію об'єкта, яку можна перемістити в необхідне місце.

6.4. Вставка об'єктів з буфера проміжного зберігання

Для вставки об'єкта з буфера необхідно попередньо вказати місце вставки, клацнувши лівою клавішею миші в передбачуваному місці вставки, а потім виконати команду **Edit / Paste** або скористатися інструментом  на панелі інструментів.

6.5. Видалення об'єктів

Для видалення об'єкта його необхідно попередньо виділити, а потім виконати команду **Edit / Clear** або скористатися клавішею **Delete** на клавіатурі. Слід врахувати, що команда **Clear** видаляє блок без приміщення його в буфер обміну. Однак цю операцію можна скасувати командою меню **File / Undo.**

6.6. З'єднання блоків

Для з'єднання блоків необхідно спочатку встановити курсор миші на вихідний порт одного з блоків. Курсор при цьому перетвориться на великий хрест з тонких ліній (Мал. 6.2). Тримаючи натиснутою ліву кнопку миші, потрібно перемістити курсор до вхідного порту потрібного блоку. Курсор миші прийме вид хреста з тонких здвоєних ліній (Мал. 6.3). Після створення лінії необхідно відпустити ліву клавішу миші. Свідченням того, що з'єднання створено, буде жирна стрілка біля вхідного порту блоку. Виділення лінії виробляється точно також як і виділення блоку - одинарним клацанням лівої клавіші миші. Чорні маркери, розташовані у вузлах сполучної лінії будуть говорити про те, що лінія виділена.

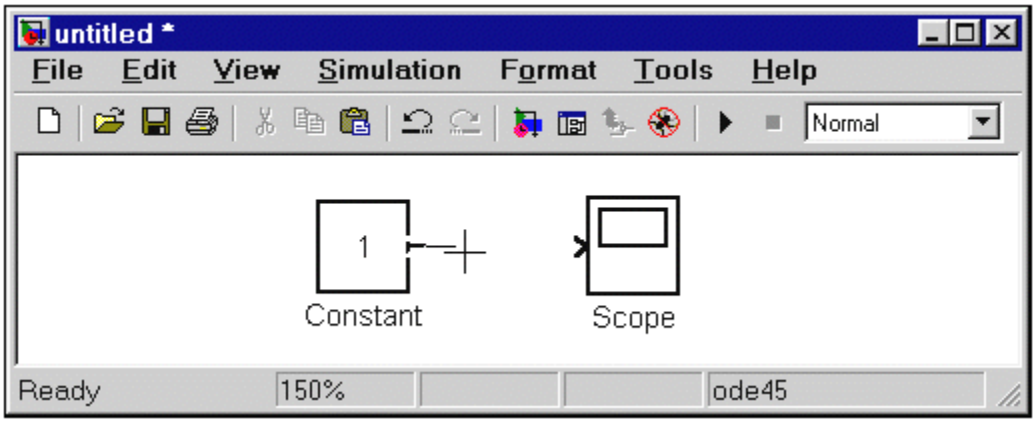


Рис 6.2. Початок створення з'єднання

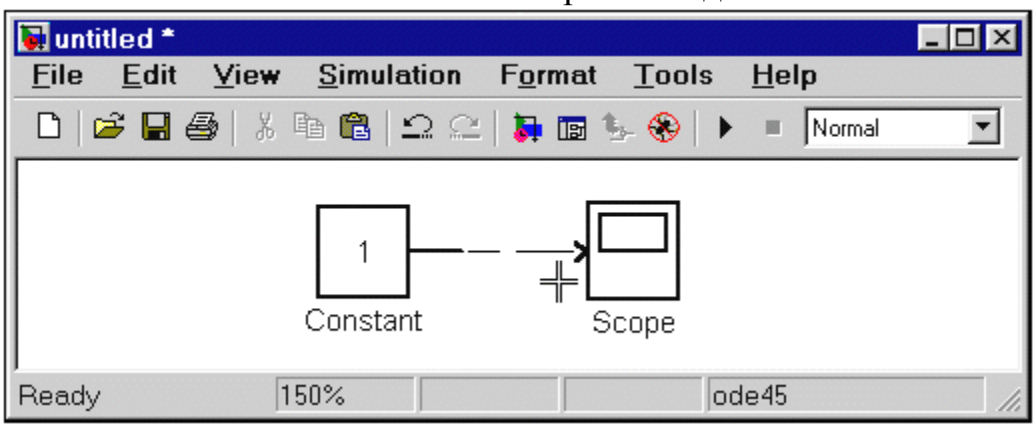


Рис 6.3. Завершення створення з'єднання

Створення петлі лінії з'єднання виконується також як переміщення блоку. Лінія з'єднання виділяється, і потім потрібна частина лінії переміщається. Малюнок 6.4 пояснює цей процес.

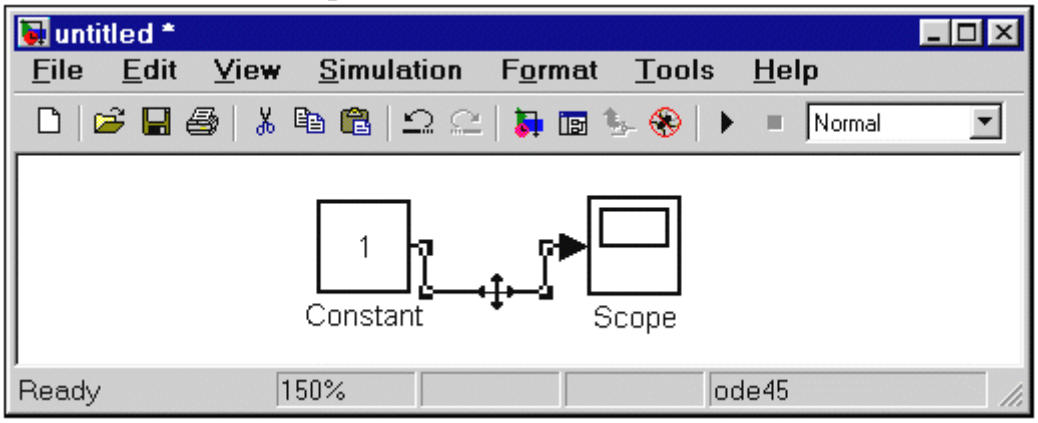


Рис 6.4. Створення петлі в сполучній лінії

Видалення сполук виконується також як і будь-яких інших об'єктів.

6.7. Зміна розмірів блоків

Для зміни розміру блоку він виділяється, після чого курсор миші треба встановити на один із маркерів по кутах блоку. Після перетворення курсору на двосторонню стрілку, необхідно натиснути ліву кнопку миші і розтягнути (або стиснути) зображення блоку. На рис. 6.5 показаний цей процес. Розміри написів блоку при цьому не змінюються.

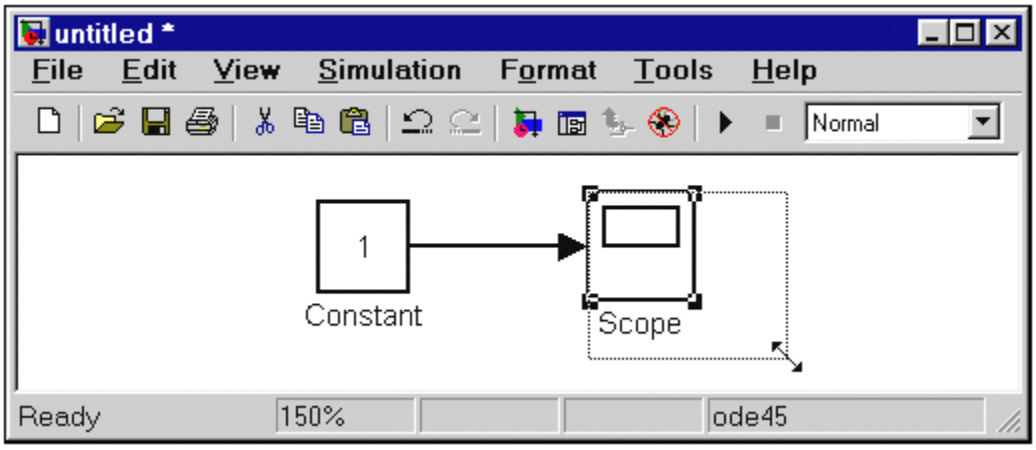


Рис. 6.5. Зміна розміру блоку

6.8. Переміщення блоків

Будь блок моделі можна перемістити, виділивши його, і пересунувши, тримаючи натиснутою ліву клавішу миші. Якщо до входів і виходів блоку підведені з'єднувальні лінії, то вони не розриваються, а лише скорочуються або збільшуються в довжині. У з'єднання можна також вставити блок, що має один вхід і один вихід. Для цього його потрібно розташувати в необхідному місці сполучної лінії.

6.9. Використання команд Undo і Redo

У процесі освоєння програми користувач може вчиняти дії удавані йому незворотними (наприклад, випадкове видалення частини моделі, копіювання і т.д.). У цьому випадку слід скористатися командою **Undo** - скасування останньої операції. Команду можна викликати за допомогою кнопки в панелі інструментів вікна моделі або з меню **Edit.** Для відновлення скасованої операції служить команда **Redo** (інструмент).

6.10. Форматування об'єктів

У меню **Format** (також як і в контекстному меню, що викликається натисканням правої клавіші миші на об'єкті) знаходиться набір команд форматування блоків. Команди форматування поділяються на кілька груп:

1. Зміна відображення написів:
   - **Font** - Форматування шрифту написів і текстових блоків.
   - **Text alignment** - Вирівнювання тексту в текстових написах.
   - **Flip name** - Переміщення підпису блоку.
   - **Show / Hide name** - Відображення або приховування підпису блоку.
2. Зміненіе квітів відображення блоків:
   - **Foreground color** - Вибір кольору ліній для виділених блоків.
   - **Background color** - Вибір кольору фону виділених блоків.
   - **Screen color** - Вибір кольору фону для всього вікна моделі.
3. Зміна положення блоку і його види:
   - **Flip block** - Дзеркальне відображення щодо вертикальної осі симетрії.
   - **Rotate block** - Поворот блоку на 90^0 за годинниковою стрілкою.
   - **Show drop shadow** - Показ тіні від блоку.
   - **Show port labels** - Показ міток портів.
4. Інші установки:
   - **Library link display** - Показ зв'язків з бібліотеками.

- **Sample time colors** - Вибір кольору блоку індикації часу.
- **Wide nonscalar lines** - Збільшення / зменшення ширини нескалярних ліній.
- **Signal dimensions** - Показ розмірності сигналів.
- **Port data types** - Показ даних про тип портів.
- **Storage class** - Клас пам'яті. Параметр, встановлюваний при роботі **Real-Time Workshop.**
- **Execution order** - Висновок порядкового номера блоку в послідовності виконання.

7. Установка параметрів розрахунку і його виконання

Перед виконанням розрахунків необхідно попередньо задати параметри розрахунку. Завдання параметрів розрахунку виконується в панелі управління меню **Simulation / Parameters.** Вид панелі управління наведено на Рис.7.1.

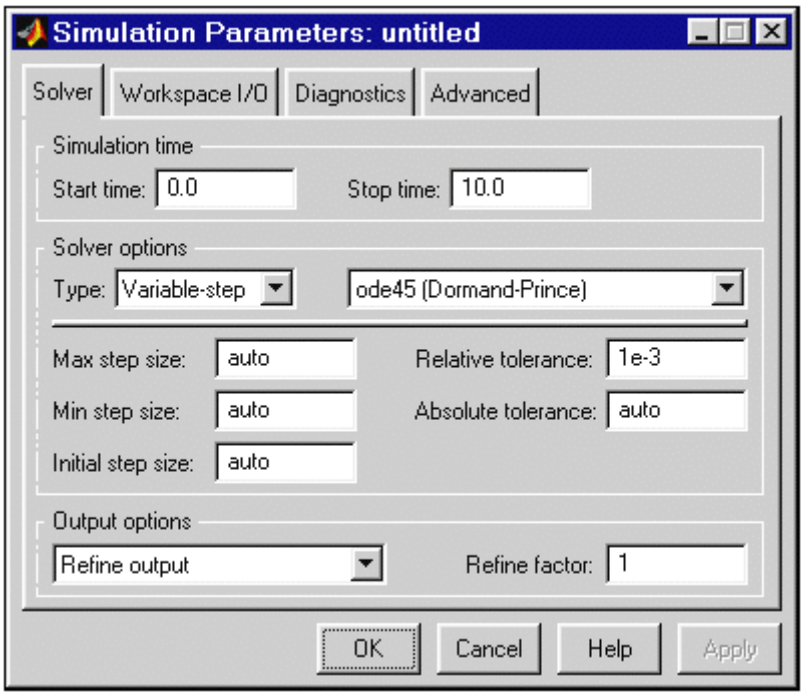


Рис 7.1. Панель управління

Вікно налаштування параметрів розрахунку має 4 вкладки:

- **Solver (Розрахунок)** - Установка параметрів розрахунку моделі.
- Workspace I / O **(Введення / виведення даних у робочу область)** - Установка параметрів обміну даними з робочою областю **MATLAB.**
- **Diagnostics (Діагностика)** - Вибір параметрів діагностичного режиму.
- **Advanced (Додатково)** - Установка додаткових параметрів.

Установка параметрів розрахунку моделі виконується за допомогою елементів управління, розміщених на вкладці **Solver.** Ці елементи розділені на три групи (рис. 7.1): **Simulation time** (Інтервал моделювання або, іншими словами, час **розрахунку), Solver options** (Параметри розрахунку), **Output options** (Параметри виводу).

7.1. Установка параметрів розрахунку моделі

7.1.1. Simulation time (Інтервал моделювання або час розрахунку)

Час розрахунку задається вказівкою початкового **(Start time)** і кінцевого **(Stop time)** значень часу розрахунку. Початковий час, як правило, задається рівним

нулю. Величина кінцевого часу задається користувачем виходячи з умов розв'язуваної задачі.

7.1.2. Solver options (Параметри розрахунку)

При виборі параметрів розрахунку необхідно вказати спосіб моделювання **(Type)** і метод розрахунку нового стану системи. Для параметра **Type** доступні два варіанти - с фіксованим **(Fixed-step)** або з перемінним **(Variable-step)** кроком. Як правило, **Variable-step** використовується для моделювання безперервних систем, а **Fixed-step** - для дискретних.

Список методів розрахунку нового стану системи містить кілька варіантів. Перший варіант **(discrete)** використовується для розрахунку дискретних систем. Решта методи використовуються для розрахунку безперервних систем. Ці методи різні для змінного **(Variable-step)** і для фіксованого **(Fixed-step)** кроку часу, але, по суті, являють собою процедури вирішення систем диференціальних рівнянь. Детальний опис кожного з методів розрахунку станів системи приведено у вбудованій довідковій системі **MATLAB.**

Нижче двох розкривних списків **Type** знаходиться область, вміст якої змінюється залежно від вибраного способу зміни модельного часу. При виборі **Fixed - step** в даній області з'являється текстове поле **Fixed-step size** (величина фіксованого кроку) дозволяє вказувати величину кроку моделювання (див. рис. 7.2). Величина кроку моделювання за замовчуванням встановлюється системою автоматично **(auto).** Необхідна величина кроку може бути введена замість значення **auto** або у формі числа, або у вигляді обчислюваного виразу (те ж саме відноситься і до всіх параметрів встановлюються системою автоматично).

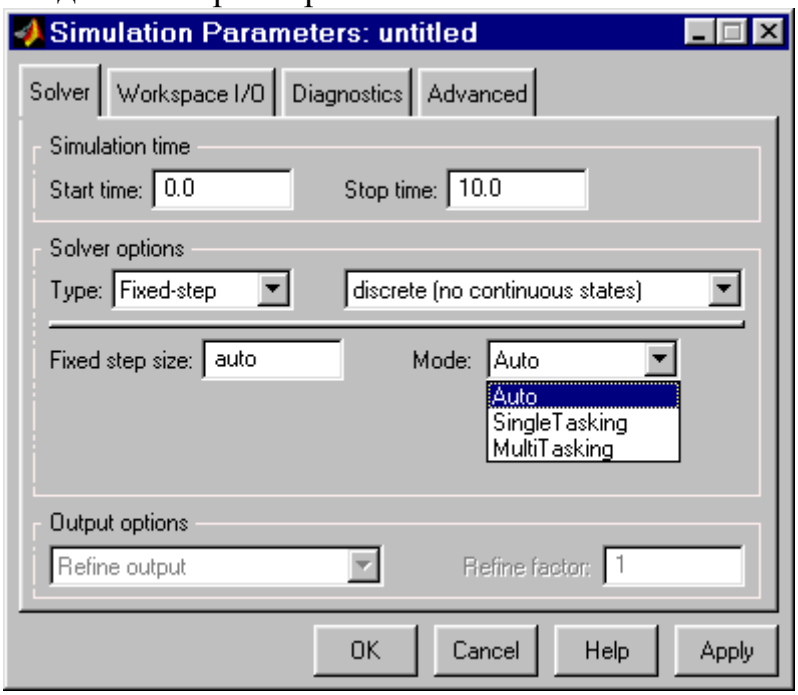


Рис 7.2. Вкладка **Solver** при виборі фіксованого кроку розрахунку

При виборі **Fixed - step** необхідно також задати режим розрахунку **(Mode).** Для параметра **Mode** доступні три варіанти:

- **MultiTasking** (Багатозадачний) - необхідно використовувати, якщо в моделі присутні паралельно працюючі підсистеми, і результат роботи моделі залежить від часових параметрів цих підсистем. Режим дозволяє

виявити невідповідність швидкості і дискретності сигналів, що пересилаються блоками один одному.

- **SingleTasking** (однозадачних) - використовується для тих моделей, в яких недостатньо сувора синхронізація роботи окремих складових не впливає на кінцевий результат моделювання.
- **Auto** (Автоматичний вибір режиму) - дозволяє **Simulink** автоматично встановлювати режим **MultiTasking** для тих моделей, в яких використовуються блоки з різними швидкостями передачі сигналів і режим **SingleTasking** для моделей, в яких містяться блоки, які оперують однаковими швидкостями.

При виборі **Variable-step** в області з'являються поля для встановлення трьох параметрів:

- **Max step size** - максимальний крок розрахунку. За замовчуванням він встановлюється автоматично **(auto)** і його значення в цьому випадку дорівнює **(SfopTime - StartTime) / 50.** Досить часто це значення виявляється занадто великим, і спостережувані графіки являють собою ламані (а не плавні) лінії. У цьому випадку величину максимального кроку розрахунку необхідно задавати явно.
- **Min step size** - мінімальний крок розрахунку.
- **Initial step size -** початкове значення кроку моделювання.

При моделюванні безперервних систем з використанням змінного кроку необхідно вказати точність обчислень: відносну **(Relative tolerance)** і абсолютну **(Absolute tolerance).** За умовчанням вони рівні відповідно **10^{-3}** і **auto.**

7.1.3. Output options (Параметри виводу)

У нижній частині вкладки **Solver** задаються налаштування параметрів виводу вихідних сигналів модельованої системи **(Output options).** Для даного параметра можливий вибір одного з трьох варіантів:

- **Refine output** (Скоригований висновок) - дозволяє змінювати дискретність реєстрації модельного часу і тих сигналів, які зберігаються в робочій області **MATLAB** за допомогою блоку **Te Workspace.** Установка величини дискретності виконується в рядку редагування **Refine factor,** розташованої праворуч. За замовчуванням зна чення **Refine factor** одно 1, це означає, що реєстрація проводиться з кроком D t = 1 (тобто для кожного значення модельного часу :). Якщо задати **Refine factor** дорівнює 2, це означає, що буде реєструватися кожне друге значення сигналів, 3 - кожне третє т. д. Параметр **Refine factor** може приймати тільки цілі позитивні значення
- **Produce additional output** (Додатковий висновок) - забезпечує додаткову реєстрацію параметрів моделі в задані моменти часу; їх значення вводяться в рядку редагування (в цьому випадку вона називається **Output times)** у вигляді списку, укладеного в квадратні дужки. При використанні цього варіанта базовий крок реєстрації (D t) дорівнює 1. Значення часу в списку **Output times** можуть бути дробовими числами і мати будь-яку точність.

- **Produce specified output only** (Формувати тільки заданий висновок) - встановлює висновок параметрів моделі тільки в задані моменти часу, які вказуються в полі **Output times** (Моменти часу виводу).

7.2. Установка параметрів обміну з робочою областю

Елементи, що дозволяють управляти введенням і виведенням в робочу область **MATLAB** проміжних даних і результатів моделювання, розташовані на вкладці **Workspace I / O** (рис. 7.3).

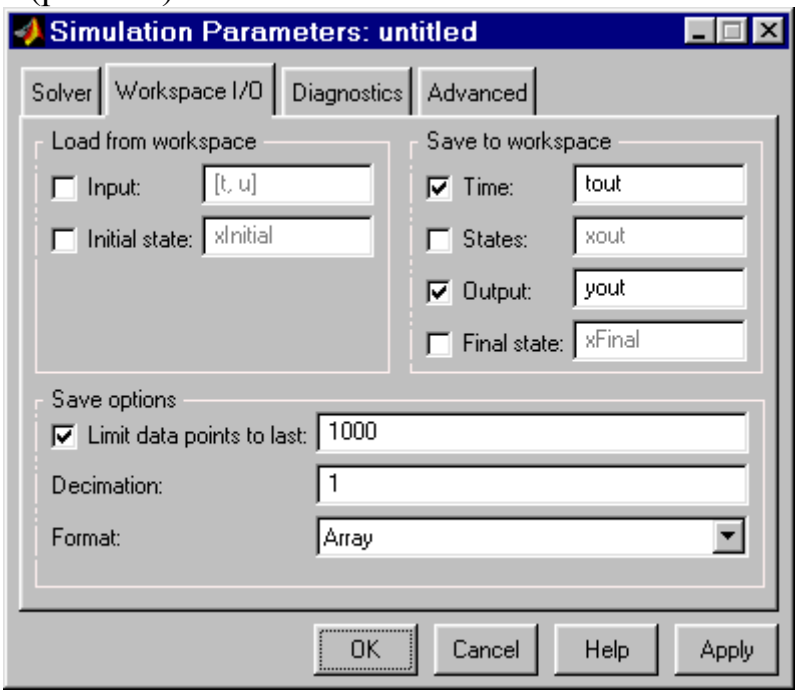


Рис 7.3. Вкладка **Workspace I / O** діалогового вікна установки параметрів моделювання

Елементи вкладки розділені на 3 поля:

- Load from workspace (Завантажити з робочої області). Якщо прапорець **Input** (Вхідні дані) встановлений, то в розташованому праворуч текстовому полі можна ввести формат даних, які будуть зчитуватися з робочої області **MATLAB.** Установка прапорця **Initial State** (Початковий стан) дозволяє ввести в пов'язаному з ним текстовому полі ім'я змінної, що містить параметри початкового стану моделі. Дані, зазначені в полях **Input** і **Initial State,** передаються у виконувану модель допомогою одного або більше блоків **In** (з розділу бібліотеки **Sources).**
- **Save to workspace** (Записати в робочу область) - Дозволяє встановити режим виведення значень сигналів в робочу область **MATLAB** і задати їх імена.
- **Save options** (Параметри запису) - Задає кількість рядків при передачі змінних в робочу область. Якщо прапорець **Limit rows to last** встановлений, то в полі введення можна вказати кількість переданих рядків (відлік рядків виробляється від моменту завершення розрахунку). Якщо прапорець не встановлений, то передаються всі дані. Параметр **Decimation** (Виняток) задає крок запису змінних в робочу область (аналогічно параметру **Refine factor** вкладки **Solver).** Параметр **Format** (формат даних) задає формат переданих в робочу область даних.

Доступні формати **Array** (Масив), **Structure (Структура), Structure With Time** (Структура з додатковим полем - "час").

7.3. Установка параметрів діагностування моделі

Вкладка **Diagnostics** (рис. 7.4) дозволяє змінювати перелік діагностичних повідомлень, що виводяться **Simulink** в командному вікні **MATLAB,** а також встановлювати додаткові параметри діагностики моделі.

Повідомлення про помилки або проблемних ситуаціях, виявлених **Simulink** в ході моделювання і вимагають втручання розробника виводяться в командному вікні **MATLAB.** Початковий перелік таких ситуацій і вид реакції на них наведено у списку на вкладці **Diagnostics.** Розробник може вказати вид реакції на кожне з них, використовуючи групу перемикачів у поле **Action** (вони стають доступні, якщо в списку обрано одне з подій):

- **None** - ігнорувати,
- **Warning** - видати попередження і продовжити моделювання,
- **Error** - видати повідомлення про помилку і зупинити сеанс моделювання.

Обраний вид реакції відображається в списку поряд з найменуванням події.

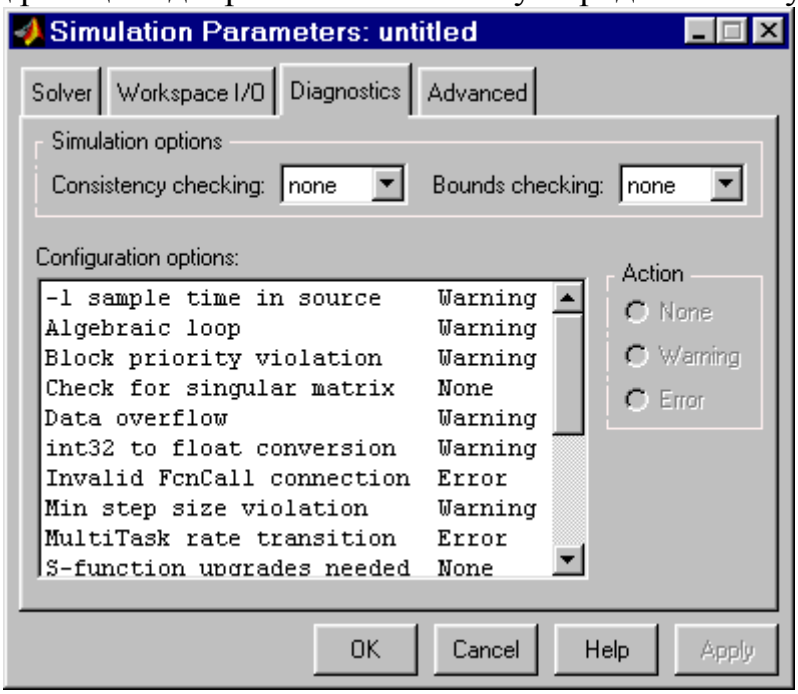


Рис 7.4. Вкладка **Diagnostics** вікна установки параметрів моделювання

7.4. Виконання розрахунку

Запуск розрахунку виконується за допомогою вибору пункту меню **Simulation / Start.** або інструмента ▶ на панелі інструментів. Процес розрахунку можна завершити достроково, вибравши пункт меню **Simulation / Stop** або інструмент ■. Розрахунок також можна зупинити **(Simulation / Pause)** і потім продовжити **(Simulation / Continue).**

8. Завершення роботи

Для завершення роботи необхідно зберегти модель у файлі, закрити вікно моделі, вікно оглядача бібліотек, а також основне вікно пакета **MATLAB.**

9. Вказівки по підготвці і проведенню лабораторної роботи:

1.Вивчити теоретичний матеріал, що відноситься до роботи.

2.Отримати у викладача завдання відповідно до свого варіанту:
3.Оформити звіт по виконаній роботі.

10. Зміст звіту.

Звіт повинен включати: мету роботи; основні теоретичні відомості і розрахункові формули; завдання, отримане у викладача; результатати моделювання. У висновках дати оцінку результатів моделювання.

Лабораторна робота №6.
Тема. МОДЕЛЮВАННЯ ПОСЛІДОВНОСТІ АВТОРЕГРЕСІЇ І КОВЗНОГО СЕРЕДНЬОГО

Мета роботи. Моделювання послідовностей авторегресії і ковзного середнього, набуття практичного досвіду моделювання процесів авторегресії і ковзного середнього.

1. Основні теоретичні відомості.

В результаті вичення чи спостереження біомедичних, акустичних, соціологічних та інших явищ і процесів стохастичного характеру отримуєм певні результати, що в ряді випадків можуть бути представлені у вигляді вектора

$$x = (x_1, ..., x_i, ..., x_n), \quad (1)$$

де $x_i(t_i)$ - результат спостереження в момент часу t_i. Вектор (1) часто розглядають як скінченну реалізацію послідовності випадкових величин

$$\xi = (\xi_1, ..., \xi_i, ..., \xi_n), \quad (2)$$

що задовільняє тим чи іншим умовам (стаціонарність послідовності; випадкової величини послідовності незалежні; функція розподілу випадкової величини $\xi_i, i = \overline{1,n}$, певний вид і т.д.).

В залежності від послідовності (2) і поставлених задач для аналізу результатів спостереження використовують різні методи, використовуючи при цьому спектрально – кореляційну теорію, метод характеристичних функцій і т.д.

Але часто виникає потреба побудувати конструктивну модель досліджуваного процесу з метою відобразити в ній механізм породження, закономірність утворення членів послідовності (2) у вигялді певного рівняня, враховуючи причинно – наслідкові зв'язки. Зафіксувати в моделі ті чи інші закони, що їм задовільняють ймовірнісні характеристики елементів послідовності (2) і т.д. Наявність такої моделі в ряді випадків спрощує оцінювання спектрально – кореляційних характеристик процесу, функції передачі системи, побудову прогнозу спостережуваного процесу та інше. Прикладами таких моделей є послідовність авторегресії і ковзного середнього. При їх означенні використовується білий шум

$$(\eta_n), \quad (3)$$

що являє собою послідовність незалежних і одинаково розподілених випадкових величин. В (3) індекс n може приймати скінченне або нескінченне значення.

Послідовність ковзного середнього. Послідовність

$$\xi_n = \sum_{k=-\infty}^{\infty} \alpha_k \cdot \eta_{n-k}\,, \tag{4}$$

де α_k -в загальному випадку комплексні числа такі, що

$$\sum_{k=-\infty}^{\infty} |\alpha_k|^2 < \infty\,,$$

називається послідовністю, утвореною з допомогою (двостороннього) ковзного середнього із послідовності (3). Послідовність (4) є стаціонарною.
В частинному випадку, коли всі α_k з від'ємними індексами рівні нулю, тобто

$$\xi_n = \sum_{k=0}^{\infty} \alpha_k \cdot \eta_{n-k} \tag{5}$$

послідовність $\{\xi_n\}$ називається послідовністю одностороннього ковзного середнього. Якщо всі $\alpha_k = 0$ при $k < m$, тобто якщо

$$\xi_n = \alpha_0 \eta_n + \alpha_1 \eta_{n-1} + \ldots + \alpha_m \eta_{n-m}\,, \tag{6}$$

то послідовність $\{\xi_n\}$ називається послідовністю ковзного середнього порядку m.

Моделі (4)-(6) дозволяють записати в явному виді їх певні ймовірнісні характеристики: кореляційну функцію, спектральну густину і інше.

Послідовність ковзного середнього. Послідовність

$$\xi(\omega, n) = \sum_{j=-\infty}^{\infty} \varphi_j \cdot \zeta(\omega, n-j), \quad \omega \in \mathbf{\Omega}, \quad n \in \mathbf{Z}\,, \tag{7}$$

де $\{\varphi_j, \quad j \in \mathbf{Z}\}$ - у загальному випадку комплексні числа, які задовольняють умові

$$\sum_{j=-\infty}^{\infty} |\varphi_j|^2 < \infty\,, \tag{8}$$

називається послідовністю, утвореною за допомогою (двостороннього) ковзного середнього із послідовності.

В частинному випадку, коли всі φ_j з від'ємними індексами рівні нулю, тобто

$$\xi(\omega,n)=\sum_{j=0}^{\infty}\varphi_j\cdot\zeta(\omega,n-j),\quad \omega\in\mathbf{\Omega},\quad n\in\mathbf{Z} \tag{9}$$

послідовність (9) називається послідовністю одностороннього ковзного середнього. Якщо у випадковій послідовності (7) всі $\varphi_j=0$ при $j>M$, тобто якщо

$$\xi(\omega,n)=\sum_{j=0}^{M}\varphi_j\cdot\zeta(\omega,n-j),\quad \omega\in\mathbf{\Omega},\quad n\in\mathbf{Z}, \tag{10}$$

то послідовність (10) називається послідовністю ковзного середнього порядку M.

Моделі (8), (9), (10) відносяться то так званих конструктивних математичних моделей стохастичних сигналів і дозволяють записати в явному вигляді їх певні імовірнісні характеристики: кореляційну функцію, спектральну щільність та інші.

Випадкова послідовність авторегресії. Випадкова послідовність є послідовністю авторегресії порядку P, якщо її можна подати у вигляді

$$\xi(\omega,n)=\sum_{k=1}^{P}\alpha_k\cdot\xi(\omega,n-k)+\zeta(\omega,n),\quad \omega\in\mathbf{\Omega},\quad n\in\mathbf{Z}, \tag{11}$$

де $\left\{\alpha_k,\ k=\overline{1,P}\right\}$ - у загальному випадку комплексні числа, які називаються коефіцієнтами авторегресії. Математичне сподівання дискретного білого шуму в конструкції послідовності (64) $\mathbf{M}\{\zeta(\omega,n)\}=0$, а дисперсія $\mathbf{M}\{\zeta^2(\omega,n)\}=\sigma_\zeta^2$. Якщо корені комплексного характеристичного рівняння $Z^P+a_1Z^{P-1}+\ldots+a_P=0$ лежать всередині одиничного кола, то рівняння (11) має одиничний стаціонарний розв'язок, який рівний послідовності одностороннього ковзного середнього (9).

Можна показати, що між послідовністю коефіцієнтів $\left\{\alpha_k,\ k=\overline{1,P}\right\}$ рівняння (11) і послідовністю значень ядра $\{\varphi_j, j\in\mathbf{Z}\}$ процесу (9) існують наступні залежності:

$$\begin{gathered}\alpha_0\varphi_0=1,\\ \sum_{i=0}^{s}\alpha_i\varphi_{s-i}=0,\ s=\overline{1,P-1},\\ \sum_{i=0}^{P}\alpha_i\varphi_{s-i}=0,\ s=P,P+1,\ldots.\end{gathered} \tag{12}$$

Із (12) безпосередньо витікають рекурентні співвідношення, зручні для реалізації на ЕОМ:

$$\alpha_0 = 1,$$
$$a_s = -\sum_{i=1}^{s} \varphi_i \alpha_{s-i}\ ,\ s = \overline{1, P-1},$$
$$a_s = -\sum_{i=1}^{P} \varphi_i \alpha_{s-i}\ ,\ s = P, P+1, ... \qquad (13)$$

Легко побачити, що формули (13) зворотні і можуть мати вигляд:

$$\varphi_0 = 1,$$
$$\varphi_s = -\sum_{i=1}^{s} \alpha_i \varphi_{s-i} \quad s = \overline{1, P-1},$$
$$\varphi_s = -\sum_{i=1}^{P} \alpha_i \varphi_{s-i}\ ,\ s = P, P+1, ...\ . \qquad (14)$$

Співвідношення (12) - (14) є основою при переході між моделями випадкового процесу у вигляді ковзного середнього та авторегресії.

Основним інструментом у моделюючому алгоритмі є система рівнянь Юла-Уокера, яка утворюється шляхом домноження на ξ_{n+s} двох частин рівняння (11) і взяття математичного сподівання, що приводить до рекурентної залежності:

$$\sum_{i=0}^{P} a_i R_{s-i} = 0, \qquad s = 1, 2, 3, ...,\ a_0 = 1, \qquad (15)$$

де R_s - кореляційна функція процесу (11):

$$R_s = \sigma_\zeta{}^2 \sum_{j=0}^{\infty} \varphi_j \varphi_{j+s}\ . \qquad (16)$$

Оскільки нам необхідно обчислювати послідовність коефіцієнтів $\{a_i, i = \overline{1, P}\}$ рівняння авторегресії (11), то відношення (12) подамо у вигляді:

$$a_1 R_0 + a_2 R_1 + ... + a_P R_{P-1} = -R_1$$
$$a_1 R_1 + a_2 R_0 + ... + a_P R_{P-2} = -R_2$$
$$\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots...$$
$$a_1 R_{P-1} + a_2 R_{P-2} + ... + a_P R_0 = -R_P\ .$$

Елементи R_{ij} кореляційної матриці порядку P

$$\begin{Vmatrix} R_0 & R_1 & ... & R_{P-1} \\ R_1 & R_0 & ... & R_{P-2} \\ ... & ... & ... & ... \\ R_{P-1} & R_{P-2} & ... & R_0 \end{Vmatrix} \qquad (17)$$

залежать від різниці індексів i-j, відповідно матриця (17) є теплицевою. Для теплицевих матриць розроблені ефективні числові методи, зокрема, для розв'язання систем лінійних рівнянь.

Моделюючий алгоритм. На основі наведених міркувань алгоритм цифрового моделювання реалізацій дискретного стаціонарного лінійного процесу (10) полягає в наступному.

1. Оцінюються коефіцієнти $\{a_i, i=\overline{1,P}\}$, рівняння авторегресії (11) шляхом розв'язку системи рівнянь Юла-Уокера (15) по заданій кореляційній матриці (17).

2. Оцінюється послідовність ядер $\{\varphi_j, j=\overline{1,k}\}$ лінійного процесу (10) по рекурентним співвідношенням (14) і послідовності коефіцієнтів $\{a_i, i=\overline{1,P}\}$, врахувавши, що $a_0\varphi_0=1$. Проводиться нормування отриманої множини відліків ядра $\{\varphi_j, j=\overline{0,k}\}$ за формулою

$$\hat{\varphi}_j=\frac{\varphi_j}{\sqrt{\sum_{j=0}^{k}\varphi_j^2}},\quad j=\overline{0,k}. \tag{18}$$

3. Для подальшого порівняння з вихідною кореляційною функцією обчислюється оцінка кореляційної функції процесу (10) згідно формули

$$R_s=\sigma_\zeta{}^2\sum_{j=0}^{k-s}\hat{\varphi}_j\hat{\varphi}_{j+s},\quad s=\overline{0,P}. \tag{19}$$

4. Визначається оцінка дисперсії за послідовністю коефіцієнтів $\{\varphi_j, j=\overline{0,k}\}$ для контролю обрахунків за формулою:

$$\sigma_\xi^2=\sigma_\zeta^2\sum_{j=0}^{k}\hat{\varphi}_j{}^2. \tag{20}$$

5. Визначається математичне сподівання білого шуму за формулою

$$m_\zeta=\frac{m_\xi}{\sum_{j=0}^{k}\hat{\varphi}_j}. \tag{21}$$

6. Генеруються реалізації дискретного стаціонарного білого шуму (тип закону розподілу може бути довільним) з математичним сподіванням (21) та дисперсією $\sigma_\zeta^2=\sigma_\xi^2\Big/\sum_{j=0}^{k}\hat{\varphi}_j{}^2$.

7. Моделюються реалізації лінійного дискретного випадкового процесу $\{\xi(\omega,n),\ \omega\in\mathbf{\Omega},\ n=\overline{0,N}\}$ згідно формули

$$\xi_\omega(n)=\sum_{j=0}^{K}\varphi_j\zeta_\omega(n-j). \tag{22}$$

Розглянутий метод моделювання реалізацій дискретних лінійних випадкових процесів дозволяє проводити комп'ютерне моделювання стаціонарних випадкових процесів із заданим математичним сподіванням та

заданій кореляційній функції з достатньою для практики точністю. На його основі можна розв’язувати широкий клас задач статистичного моделювання, зокрема він може бути використаний при розв’язанні задач аналізу лінійних та нелінійних перетворень стаціонарних випадкових процесів у динамічних системах, що реалізується на базі методу статистичних випробувань.

2. Основні алгоритми і розрахункові формули:

Алгоритм моделювання приведений в теоретичних відомостях, а основними розрахунковими формулами є (5) – (7).

3. Теоретичні відомості, необхідні при підготовці до проведення роботи.

1.Процеси Маркова. Ланцюги Маркова. Матриця переходу.

2.Алгоритм побудови ланцюга Маркова.

4. Вказівки по підготвці і проведенню лабораторної роботи:

1.Вивчити теоретичний матеріал, що відноситься до роботи.
2.Отримати у викладача завдання відповідно до свого варіанту:
3.Провести на ЕОМ моделювання ланцюга Маркова.
4.Оформити звіт по виконаній роботі.

5. Зміст звіту.

Звіт повинен включати: мету роботи; основні теоретичні відомості і розрахункові формули; завдання, отримане у викладача; результатати моделювання. У висновках дати оцінку результатів моделювання.

6. Контрольні питання.

1.Визначення марківського ланцюга.
2.Стани системи і матриця переходів.
3.Алгоритм отримання ланцюга Маркова.

ЛІТЕРАТУРА:

1. Основы системного анализа и проектирования АСУ: Учеб. пособие /А.А. Павлов, СН. Гриша, В.Н. Томашевский и др.; Под общ. ред. А.А. Павлова. - К. Выща шк,, 1991. -367 с.
2. Томашевський В.М. Імітаційне моделювання систем і процесів: Навч. посібник. - К.: ІСДО, 1994.-124 с.
3. Советов Б.Я., Яковлев С.А. Моделирование систем: Учеб. для вузов по спец. "Автоматизированные системы управления". - М.: Высш. шк., 1985.- 271 с.
4. Бусленко Н.П. Моделирование сложных систем. М.: Наука, 1978. - 400 с.
5. Прицкер А. Введение в имитационное моделирование и язык СЛАМ П.- М.: Мир, 1987.- 646 с.
6. Томашевский В.Н., Жданова Е.Г. Имитационное моделирование средствами GPSS. : Учеб. пособие. К.: ІЗМН, КГП, 1998. - 123 с.
7. Томашевський В.Н., Жданова О.Г., Жолдаков О.О. Вирішення практичних завдань методами комп'ютерного моделювання: Навч. посібник. - К.: "Корнійчук", 2001. — 268 с.
8. Томашевский В.Н. Моделювання систем і процесів: Електронний навч. посібник. К.: Кафедра "АСОІУ НТУУ КПГ, 2001.
9. Методы построения имитационных систем / Литвинов В.В., Марьянович Т.П. - К.: Наукова думка. 1991. - 120 с.